الجمهورية العربية السورية
جامعة حلب
وزارة التعليم العالي
كلية الهندسة الكهربائية والالكترونية
قسم التحكم الالي والأتمتة

التحكم باستخدام انظمة الهواتف الذكية







مشروع اعد لنيل الاجازة في كلية الهندسة الكهربائية والالكترونية

قسم التحكم الألي و الأتمتة

باشراف
الدكتور المهندس عماد الروح
اعداد الطلاب
ياسر كركوكلي
مؤيد شيخ امين

للعام الدراسي 2015 /2016

الفهرس

الفهرس:

الفهرس:

الفهرس:

كلمة شكر

(هل جزاءُ الإحسانِ إلا الإحسانْ)

أقدر دور من حولي ***أزيد بحقهم وعياً

بلادي أمتي أهلي * فلستُ بدونهم شيئاً**

أقف على منبر قلبي ، وأنادي من محراب روحي ، وأدعو ربي ليجزي عني خير الجزاء ، كل من علَّمني حرفاً ، أو أعانني طرفاً ، أو قدَّر لي ظرفاً .

ومعلمٌ وكفى بكلّ مُعلّمٍ شرفاً * وقد بُعثَ النبيُّ مُعلِّماً**

طِفتُ في عالم الفضل أنظر الأفضال التي غمرتني من كل حدبٍ وصوب ، فوجدت أنَّ ذكر أهل الفضل تضيق عنه القراطيس والدواوين فغدوت أخط على لوحي هذا كُلَّ اسمٍ لاح لي في هذا الوقت فإن تاه العقل عن ذي فضلٍ فإنَّ القلب لا يتوه .

فأول كلمة شكر أرفعها إلى أساتذتي الفاضلين الذين حملوا رسالة المُعلِّم الأكبر سيدي وحبيبي محمد (ﷺ) وأقول لهم :

دامَ فضلكم وشُكِر سعيكم وجُزيتم عني كُلَّ خير أساتذتي :

الدكتور المهندس رئيس قسم التحكم و الأتمتة بجامعة حلب (عماد الروح) الذي مهَّد الطريق ليرى النورَ مشروعنا وشرَّفنا بأن كان اسمه وساماً ازدانت به أطروحتنا وأنار لنا دروب العلم بمشكاة علمه على مدى ثلاث سنوات جامعية .

ثُمَّ أتقدم بالشكر موصولاً إلى بقية أساتذتي ممن كانوا سبباً سواء بمراجعهم أو دروسهم التدريبية أو علمهم الغزير في القوة العلمية التي شجعتني على الإقدام لخوض غمار هذا النمط الجديد من دراسات التحكم والاختصاص .

المهندس صفوان أبو جوهر ، المهندس عبد الكريم ، المهندس هاشم المهندس ، المهندس أحمد أشرم ، المهندس توماس صاووق ، المهندس بدر الدين ريحاوي ، المهندس وليد بليد ، المهندس عبيدة بدوي ، المهندس أحمد حسكل ، أخي المهندس وليد عيسى ، أخي المهندس معاذ أبو طعمة ، أستاذي أحمد قضيماتي ،أستاذي كامل غنوم ، معلمتي لمى (رحمها الله) .

ثُم أعقبه امتناناً لكل من ساهم معي ولو بلحظة أو كلمة في هذا المشروع من أصدقائي :

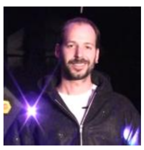
قسم هندسة الحواسيب: يمان قلعة جي ، طه سليمان ، عبد الله سفلو

قسم هندسة الإلكترون : محمد رمضان ، رشيد بوزان.

قسم هندسة المعلوماتية : عبد القادر الخطيب

قسم هندسة التحكم والأتمتة : وائل كويفاتي ، مصطفى بشير ، محمود حج حسين ، فاتح كادورة ، حسن شوفان .

وإلى الذين ساعدوني من مطوري تطبيقات تحكم بالأندرويد :

الأميركي Ytai Ben-Tsvi  والتايلاندي ذو الأصل الياباني :MASTER UNG THAI 

الدكتور الأميركي : Simon Monk والآنسة المُطوِّرة الاسبانية بقناتها التدريبية المميزة في يوتيوب : VIVANA Roze

الإهداء

إليكَ يا مَن بِحُبك عرفتُ نفسي وبدينك رفعتُ رأسي وبطاعتك شرُفَ قدري وبرضاكَ تحقَّقَ أُنسي فأنت إن ترضى فإني لا أُبالي بالمصير فَـ

يا حبيبَ القلوب مَن لي سواكا * ارحمِ اليومَ مُذنباً قد أتاكـــا**

أنتَ سؤلي ومُنيَتِي وسروري * قد أبى القلبُ أن يُحبَّ سواكا**

ليس سؤلي من الجنانِ نعيمٌ * غيرَ أنّي أُريــدها لأراكـــا**

رباه يا الله.

إنْ كانَ أحببتُ بعدَ اللهِ مثلهُ في * بدوٍ وحضرٍ ومِن عُربٍ ومِن عجمِ**

فلا اشتفى ناظري من منظرٍ حسنٍ * ولا تفوَّه بالقول الســـديدِ فمي**

إلى من له الشفاعة الكُبرى والمنزلة العظمى والحوض المورود والمقام المحمود واللواء المعقود سيدي وحبيبي

محمدٌ رسول الله ﷺ .

ربَّى رجالاً صاروا قياصرةً * من بعدِ أن كانوا رعاة الشاة والغنمِ**

إلى أجدادنا الذين من سار على سُنَّتهم أفلحَ واهتدى ومن تركها فقد ضلَّ والمهالكَ ارتدى.

((الخلفاء الراشدون : الصديق والفاروق وذو النورين ومَن كَرَّمَ الله وجهه رضي الله عنهم ورضوا عنه))

و من للمصطفى صحبوا فخرُ ماضينا المجيد ((صحابة النبي وآله رضوان الله عليهم أجمعين))

إلى المسجد الأقصى الأسير أهديه إنجازي وأُبشرهُ بقولي :

لا تبكِ أقصانا بل ارتقب * واشهد فينا صلاحَ دينٍ يثِبْ**

إلى أُمتي الإسلامية وبلادي بلاد الشام (تاركاً لها بصمتي العلمية) إليك شهباءُ يا مرتع طفولتي التي شداك جدي ((رحمه الله)) بقوله :

مِنَ الشهباءِ منبَتُ كُلِّ وُدٍّ * تربَّع فوقَ سُـــدَّتها الرزانُ**

مجالسها يسودُ بها التآخي * وفي أرجائها حبٌ ، حنانُ**

ويغشاها الوفاءُ بكُلِّ حيِّ * ثِقاةٌ أهلُها ظلُّوا وكـانـــوا**

مغاني بهجةٍ ورياضُ جودٍ * يُشيرُ إلى شمائلها البنـانُ**

ديارٌ باركَ الرحمنُ فيها * وحيَّتها الملائكة الحســـانُ**

وقلعتُها شموخٌ بالمعالي * لمجدِ الفاتحينَ هي البيـــانُ**

إلى مرتَعِ طفولتي ومصنع رجولتي دُرَّة الوجود وبحر الجود نبراس العُلا المرفود بالحُبِّ والخير الممدود ، إلى التي أسمتني ياسرَ صعبِ ذا الزمانِ فكانت لحياتيَّ (الدنيا والآخرة) سبب نجاتي ، إلى التي قُلتُ في فضلها وإحسانها :

أيَا أُمَّاه لولاكـــي * لَما بُنيَت أنفاسي**

وبرضاكي أيا أُمَّاه * لقد حققتُ أهدافي**

أمي الحبيبة الفاضلة

يامَن بماء رعايتكَ أينعَ ثمري ، نصبت إلى التعبِ جسراً ، وخُضتَ مركِبَ البذلِ بحراً ،وهجرت من أجلي الرُقادَ يوماً فدهراً ، بمطلعِ كلِّ فجر أدعوا الله أن يحفظك وأمِّي ، ويعينني على إيفاء ما استطعت من يسيرِ حقِّ البِرِّ .

إليك أبتِ

معبراً عن شعوري * وفـرحتي وامتناني**

لِمَنْ لـــي عـــونٌ * على عوادي الزمانِ**

ومن هم ليَ أُنسٌ * في وحشــــةِ الأحزانِ**

هُم بواعثُ فخري * و عِزَّتي وافتتــاني**

بِمثلهم يتبـــاهى * أخٌ بكلِّ مكـــــــانِ**

إلى إخوتي(عامر، عبد الرحمن ، حذيفة)

إلى شمعةِ حياتي ونور فؤادي ، مهجة قلبي وقرَّة عيني والنعمة المحمودة المُزجاة من ربي

(أختي ناهد)

قيل (رُبَّ أخٍ لم تلده لك أُمُّك ورُبَّ أختِ كذلك) فلذلك أوصل شكري وامتناني

لإخوتي ((عمر ، عبد الله ، معاذ ،حكمت ،جودت، مؤيد، لطفي، أحمد وابراهيم، أستاذ مصباح))

وأخواتي((شهد ، نديمة ،غدير))

إلى مَن كان لي قنديلاً نبراساً وقدوةً مُثلى بهذا الزمان ، فكان يكلؤني حُباً وحناناً وسقاني بفيء علمه حكمةً ، رجولةً ، إخلاصاً ، ثباتاً وضياءً ورعاني مؤدباً موجها للحق والصراط المستقيم ، إلى مّن أذكُر أياماً لي معه خَلَتْ فيلتهب قلبي شوقاً له ويذوبُ كمداً ويزيدُ فيه الوجد اتقاداً .

جدِّي الفاروق عمر رحمه الله ورضي عنه

إلى جوهرة العائلة وجمانة حُبِّها منبع الحب والاهتمام، صاحبة القلب الحنون الذي لا ينام ، أعظم جدة بهذا الزمان

حبيبتي وجدتي نديمة

إلى أمي الثانية الكبرى إلى ثرى روحك الطاهرة يا مّن أنتِ دون الخلائق كُلِّها أردتُ فرحتك وبسمتكِ حين تخرجي لكن أبت يدُ المنيّةِ إلا أن تمضي بك إلى الرحمن شهيدةً هانئةً سعيدة راضيةً مرضية فحُقَّ ارتقابك في قولِ أبيكِ رحمه الله :

وعلى دروبِ الفضلِ سارت والتُّقى * ترنو إلى الفردوسِ بالنظراتِ**

خالتي ذكاء كركوكلي

إلى عضدي وسندي وعزوتي بين الأنام ،أخوالي ، خالاتي ، عماتي ، والأعمام وخاصة أبي الروحي وخالي الذي أكرمني فسمَّاني ، الذي قال فيه جدي ((رحمه الله)):

اقبالُ جاءَ إلـــينا** هدية المنَّـــانِ**

رفاقُهُ عرفـــوه ** حقاً فتى الفتيانِ**

لأنَّهم وجـــدوه**ذا نجدةٍ وتفـــانِ**

الشكر والإهداء:

ربَّاه فاسبغ عليه**غمائم الرضوان**

إلى من كُنَّا معاً سوياً حتَّى فُكَّ الخطام فتركناهم كارهين وصوتهم في الأذنِ دامَ، إلى رفاق الطفولة والشباب:

يزن ، فيصل ، محمد ، أحمد ، خالد ،سومر ، سامي ، فالنتينو ، نديم ، ظافر ، عثمان وابراهيم ، أيهم وهاشم ......

إلى أصدقائي في درب الدراسة الجامعية طلاب دفعتي الذين قُلتُ فيهم :

سنينُ عُمرٍ قد مضت * بنا وبكم قد حظت**

دروبُ علمٍ ماجدة * سرنا بها حتى ارتقت**

نفوسنا وأشرقت * بخيرِ صحبٍ أومَضت**

إلى من كان لي عوناً وسنداً من أول الطريقِ لمنتهاه فكانَ نِعمَ الأخ والصاحب والذي أتمنى أن أقضي ما تبقى من عمري بصحبته حتى ألِجَ لحدي فألقاه بعدها في جنَّةِ ربّي .

شريكي مؤيد شيخ أمين



إلى شهد الروح ومرام الفؤاد وسناء المحبة ، إلى التي خواطرها داعبت أجفاني فجَاد لأجلها وهو يجهلها بالشعرِ يراعُ بياني فقال :

ليسَ مِنَ المعيبِ أنْ تُحِبَّ حبيبةً * إنَّ المعيبَ في حُبّها أن تخسرا**

فامضِ في حُبِّك لا تخش قول الورى * فالحُبُّ حقٌ وحقُّهُ أنْ يُنصرا**



زوجة المستقبل //المجهولة//

بِسۡمِ ٱللَّهِ ٱلرَّحۡمَٰنِ ٱلرَّحِيمِ

((وقل ربي زدني علماً))

((ربي إني لما أنزلت إليَّ من خير فقير))

***

((والذين هاجروا في الله من بعدِ ما ظُلِمُوا لنُبَوِّئنَّهم في الدنيا حسنةً ولأجرُ الآخرةِ أكبر لو كانوا يعلمون *** الذين صبروا وعلى ربهم يتوكلون))

الآيات(٤١-٤٢) سورة النحل

صدق الله العظيم

وسِعتُ كتاب الله حفظاً وغايةً * وما ضقتُ عن آيٍ بهِ وعظــاتِ**

أَ فَيُعيينِي اليومَ وصفيِ لآلةٍ * وتنسيقِ أســــماءِ لمخترعاتـــي**

في(٢٥/١/٢٠١٦م) ياسر محمد كركوكلي



مقدمة ومدخل للكتاب:

هذا الكتاب عبارة عن خلاصة عملنا وأطروحتنا في مشروع التخرج الذي قمنا به لنيل الإجازة الهندسية في الكهرباء والالكترون لمتخصصي هندسة التحكم و الأتمتة ففيه ما وُجدَ بالأطروحة إضافةً لإضافات أخرى مميزة ارتأيت أن أكتبها فيه ليكون هذا الكتاب أول كتاب عربي يضاف للمكتبة العربية الإسلامية في مجال التحكم باستخدام أنظمة الهواتف الذكية فيكون لبنة وحجر أساس لمن أراد البدء والمضي في هذ المجال الهندسي المميز فيكون عوناً لمن أراد الخوض في هذا البحر العميق ، مركباً يشقُّ به عباب مياه المحيط الهندسي نحو التفوق والامتياز والبناء وتطوير الأمة الإسلامية .

و هو عبارة عن ممهد لاستخدام لوحات وبوردات الـIOIO والتعامل معها .

ستجدون في هذا الكتاب موجزاً عن المشروع وشرحاً مبسطاً

ثم في الفصل الأول سنتحدث عن العتاد البرمجي والبرامج ولغات البرمجة الضرورية لهذا المجال.

بعد ذلك في الفصل الثاني نتحدث عن القطع المستخدمة بمشروعنا وهناك ملحقٌ مهمٌ للفصل الثاني بعنوان المحركات الخطوية وبعض شرائح القيادة الهامة نورده .

ثمَّ في الفصل الثالث نتحدث عن أشهر طرق التحكم باستخدام أنظمة الأجهزة الذكية وأفضل الشرائح مع بعض المقارنات ونستعرض الطريقتان اللتان استخدمناهما .

وأخيراً بالفصل الرابع نورد أكواد المشروع البرمجية للمتحكم المستخدم وأجزاء مهمة من أكواد الأندرويد لمن أحب بناء التطبيقات لتكون معيناً له في الربط البرمجي والتخاطب ((لم نضع كل الكود البرمجي لتطبيق الأندرويد لأنه كما نعلم ببرمجة التطبيقات هناك كود المنصة وكود الواجهة وكود الجافا وكود السماحيات والشروط وكود الربط وهذه كلها أبسط كود سيستهلك العدد الكثير من الصفحات فارتأينا لهذا السبب أن نضع الأجزاء المهمة وأي مبرمج أندرويد عندها يبني تطبيقه بالاعتماد على الجزء المهم مما ذكرنا فلكل مبرمج طريقته ، ولسبب آخر كون عملنا كان جماعياً فلنا تحفظات لحقوق النشر لكود الأندرويد للمشروع ولأننا لا زلنا نريد تطويره تطويراً أفضل حتى يصل لأفضل ما نتمنى عندها عند أخذ براءة تطوير سننشره ونجعله متاحاً للجميع ليبنوا على أساسه أكوادهم))

بالفصل الرابع ألحقنا كود واجهة الجويستيك لقيادة المحركات عبر شريحتين من 293 أو 298 وتغيير السرعة والاتجاهات عبر استخدام تعليمات pwm لدارة اليويو.((وهو كود مهم وأساسي بنيناه لمن أراد التطوير وبالنسبة لي شخصياً بما أنني ((ياسر)) مبرمجه ومطوره فليس لي تحفظ عليه ومن الآن للمطورين كامل صلاحية التطوير عليه وهو للنشر أرجو من الله حسن الثواب وأن يكون حسنةً جاريةً لي)).

مع هذا الكتاب سألحق بعض الملفات الهامة والروابط وكل ذلك سيوجد على موقع أخي الأكبر وصديقي الدكتور (مستقبلاً) أستاذي المهندس الغزاوي المُبدع وليد عيسى .

ملاحظة مهمة أخيرة: بالمستقبل القريب بعد بضع شهور أو سنة ونصف على الأكثر إن تيسَّر ذلك سأقوم بتنزيل جزء ثاني للكتاب و دروس وكورسات لتعلم برمجة اليويو والتعامل معها على قناتي في اليوتيوب

https://www.youtube.com/channel/UCq0RbRuLe3oFb9e12vqzaUg

وعلى صفحتي بالفيس بوك أو صفحة هندسة التحكم في جامعة حلب

https://www.facebook.com/groups/micro.fpga.plc/?ref=bookmarks

وعلى موقع Eletorial إن شاء الله ما دمنا أحياء نرزق.

هذا الكتاب يخضع للرخصة الحرة مفتوحة المصدر بصورته الالكترونية مجاناً ويمكنك أن تنشره أو تُعدِّله كيفما تشاء شرط أن تذكر المصدر وسينشر أول مرة على صفحة هندسة التحكم في جامعة حلب وعلى موقع Eletorial.

يمكنك أن تستغل هذا العمل بصورة تجارية شرط الحصول على إذن مباشر من المؤلف المهندس ياسر كركوكلي https://www.facebook.com/yasser.karkoukli

إذا أعجبك الكتاب وأحببت دعم الكاتب وللحصول على المزيد من الكتب مفتوحة المصدر فلا تتردد بالتواصل معي https://www.facebook.com/yasser.karkoukli

بالنهاية ما كان من فضلٍ فمن الله وما كان من تقصير فمن نفسي نسأل الله الصفح والغفران .

أخوكم في الله

ابن الشهباء المكلومة ياسر محمد كركوكلي

بمساعدة إخوته : من حلب مؤيد شيخ أمين

من إدلب الخضراء طه سليمان ، عبد الله سفلو.

مقدمة عن المشروع وهدفه:

هدف علماء التحكم هو الحصول على دماغ الكتروني صغير يحاكي دماغ البشر في اتخاذ القرارات للحفاظ على استقرار نظام معين فمثلاً يوجد بدماغ الإنسان متحكم داخلي يحافظ على استقرار درجة حرارة جسم الإنسان عند 37° وعندما ترتفع عن هذا الحد عن هذا المتحكم يقوم بإصدار الأوامر لمسامات الجلد بالتعرق وخفض درجة الحرارة.

ولقد تطورت وسائل التحكم وشهدت قفزات نوعية بدءاً من القرن التاسع عشر عصر الكهرباء والثورة الصناعية (حيث التحكم الكلاسيكي واليدوي المعتمد على الكهرباء والريليهات والكونتاكتورات والمحركات) مرورا بالقرن العشرين عصر الالكترونيات والذرة والحواسيب حيث شهدت تطوراً ملحوظاً (مجالات الدارات الالكترونية المطبوعة ، المبدلات ، المنظمات ، الترانزستورات ثم الدارات المنطقية والمتكاملة فالمعالجات والمتحكمات الصغرية ، أجهزة الحواسيب PC) إلى عصرنا الحالي القرن الحادي والعشرين عصر الروبوتات والأنظمة والأجهزة الذكية وشرائح التطوير الذكية واللوحات التطويرية المبنية على معالجات ARMX,FPGA ، وشاشات اللمس الكهرطيسية . فمنذ الأيام الأولى لظهور الحاسوب , طرأت تغيرات كبيرة على التفاعل ما بين المستخدم والحاسوب ,بدءاً من لوحة المفاتيح والفأرة, ومرورا بشاشة اللمس وأجهزة الموبايل الذكية التي تشبه الحاسوب في ميزاتها حيث أصبح التواصل بين المستخدم وهذه الأجهزة الذكية أمرا سهلا بسبب الواجهات البسيطة ,حيث يمكن لأي شخص مهما كان عمره أن يستخدم هذه التكنولوجيا الجديدة,كما يمكن لهذه التقنية المتطورة مساعدة الأشخاص في مجالات أعمالهم

ولأن أية وسيلة تحكم تعتمد على الحاسوب ومعالجته وعلى المستخدم ومراقبته له و لأن التحكم حالياً يعتمد على كل الطرق التي تفضي للتحكم بالروبوتات المتحركة و صناعة روبوتات متحكم بها وخاصة عن بعد

حيث برزت أهميتها في أتمتة الأنظمة الصناعية المعتمدة على أذرع روبوتية متحركة و قيادة الروبوتات المتحركة في الأغراض التجسسية العسكرية و خصوصاً في الأماكن التي لا يستطيع الإنسان الوصول فعلى سبيل المثال (برز هذا الدور في المحطة النووية بفوكوشيما بعد تعرضها لأضرار فادحة إثر الزلزال المدمر وعدم مقدرة الإنسان على الدخول للإصلاح بسبب

معدل الإشعاعات المرتفع) وفي الأغراض الطبية كتسهيل وتبسيط عمل الأجهزة المساعدة للمرضى كالكرسي المتحرك عن طريق الأزرار والشاشات

كل ذلك هو ما دفعنا إلى صنع تطبيق برمجي يعمل على الأجهزة الذكية يمكن التحكم من خلاله حيث أننا اليوم لسنا مضطرين لحمل جهاز الكمبيوتر الشخصي أو المحمول أو الجلوس طوال اليوم أمام الكمبيوتر لأداء مهمة ما , فمع هذه الابتكارات التكنولوجية المستمرة أصبح من الممكن أن يحل الهاتف الذكي محل الكمبيوتر الشخصي في بعض المهام , كما أن الهواتف الذكية باتت في متناول الجميع بسبب سهولة التعامل معها وكون أسعارها اقتصادية ومقبولة.

إذ أدى التسارع الكبير في عجلة التطور التكنولوجي إلى توفر إمكانيات كبيرة و مرنة بنفس الوقت بين أيدي شريحة واسعة من الناس و كان لزاما علينا نحن مهندسي التحكم الاستفادة من هذا التطور و تسخير هذه الطاقات في علم التحكم

ومن أهم مكونات منظومات التحكم بشكل عام هي وحدات معالجة البيانات بالإضافة إلى منظومات الاتصال ، وباجتماع هاتين الخاصيتين في الأجهزة اللوحية المتطورة التي لاقت رواجا في الآونة الأخيرة نظرا لانتشار أنظمة التشغيل الحديثة مفتوحة المصدر ، قمنا باستخدام هذا التركيب للحصول على بنية تحكم عصرية ومتاحة للجميع ، مرنة بالقدر الكافي لتواكب أشكال التطور التقني الثوري القادم .

فالهدف من هذا المشروع هو :

١*- إضافة مجال بحثي دراسي جديد لم يوجد بالقسم ضمن الكلية سابقاً لدارسي علم التحكم ليواكبوا آخر التطورات المستجدة وأحدث التقنيات المتوفرة .

٢*- عمل نواة وبنية ارتكاز لأي مشروع يعتمد التحكم باستخدام أنظمة الهواتف الذكية من خلال شرح وعرض الطرق المستخدمة لذلك واعتماد أفضلها وأوفرها وأكثرها رواجاً وقابليةً للتطوير مستقبلاً .

٣*- أساس لأي مشروع ضمن تصنيف الروبوتات الذكية : "جيل الروبوت الثالث "
Intelligent robots

٤*- النهوض بأمة سيدي محمد ﷺ من جديد ليقر الله أعيننا ببزوغ فجرها القادم ويكون لنا بذلك يدٌ يرضى الله بها و ركناً من أركان الخير لأمته نحنُ خريجي المدرسة المُحمَّديَّة وطلابها.

موجز عن المشروع:

أردوينو هو مشروع مفتوح المصدر يتألف من قسمين عتادي وبرمجي. تم إطلاقه ليعطي المصممين والفنيين والمهندسين منصة نماذج عملية لتفعيل الأبحاث التصميمية وفق مكاتب برمجية مفتوحة المصدر تتعامل مع شرائحه المتعددة

أيضاً نظام الأندرويد هو النظام الأفضل للأجهزة الذكية كنظام مفتوح المصدر قابل للبرمجة والتطوير

الخبراء والهواة اليوم في كافة أنحاء العالم يستخدمونهما لإنشاء مشاريع برمجية وعتادية خاصة في مجال الروبوت والتحكم الذكي .

من خلال ربطهما معا (شرائح ولوحات الأردينو المطبوعة ، برامج وتطبيقات أنظمة الأندرويد) تستطيع بسهولة أكثر أن تبني و تطور مشاريعك الإلكترونية والبرمجية ابتداءً من المشاريع التجريبية انتهاءً بالمشاريع المعقدة.

الكلمات المفتاحية:

أردوينو

بوردات أردوينو ARDUINO BOARDS (LIKE UNO ,MEGA, DUE .....etc) وخاصة البورد

المطورة يويو OTG/ IOIO

الدارات المتكاملة والمتحكمات الصغرية

المعالجات الصغرية

البيئة التطويرية المتكاملة

المتحكم ATmega328p-pu

أندرويد

أجهزة ذكية

خيط

معالجة

زناد

Summary:

Arduino is an open source project consisting of both hardware and software. It was originally created to give designers,artists and Engineers a prototyping platform for interaction design courses, According to open-source software offices dealing with multiple segments.

Also android operating system is the best one for the smart phones as an open source system Programmable and development.

Today hobbyists and experts all over the world use them to create physical computing and programming projects ,

Especially in the field of robot and intelligent control.

Through networked together (slides and PCB Arduino Circuit , programs and applications of Android systems) more easily afford to build , So Arduino Boards and android system make it easier than ever to build ,develop handcrafted electronics projects ranging from prototypes to sophisticated gadgets.

KeyWords:

Arduino IOIO/OTG

Arduino Boards /UNO

Integrated circuits (IC) & Microcontrollers

Microprocessor

IDE: Integrated Development Environment

ATmega 328P-PU

Android , Smart phones, Thread , Process

والآن بسم الله نبدأ فصول الأطروحة الأربعة :

الفصل الاول

المكونات البرمجية

١.١- اللغة البرمجية لبناء التطبيقات JAVA:

تصنف لغات البرمجة إلى ثلاث مستويات رئيسية وهي machine language أي لغة الآلة و assembly language أي لغة التجميع أما بالنسبة للنوع الثالث من لغات البرمجة فيطلق عليه اسم High-level languages أي لغات البرمجة عالية المستوى و يختصر بـ HLL حيث تحتل JAVA لغة الجافا قمة الهرم في هذه العائلة . إن التطور السريع و المطرد في عالم البرمجيات أدى إلى الحاجة الماسة إلى تطوير لغات البرمجة التقليدية و تحويلها إلى لغات تتماشى مع متطلبات العصر.

٢.١- لماذا اللغة البرمجية JAVA:

تتميز JAVA لغة الجافا بأنها لغة ضخمة جداً تتماشى مع جميع أنظمة التشغيل المتداولة (أي أنها تعمل مع جميع أنظمة التشغيل operating systems) مثل نظام Windows XP و نظام UNIX و غيرها من أنظمة التشغيل ، كما أن JAVA لغة الجافا تعتبر التوأم الثاني للغة C++ حيث أن كلاهما نشأت من قبل شركة Sun Microsystem كما أنها تعتبر لغة عامة و شاملة للغات البرمجة الأخرى و التقليدية لا يمكن الإحاطة بها بالكامل إلا بالتخصص فيها لمدة ٧سنوات على الأقل ولقوة هذه اللغة وضخامتها صنفت الأولى عالمياً .

لقد أدى ظهور JAVA لغة الجافا عام 1990 إلى حل الكثير من المشاكل و أهم هذه المشاكل هي :

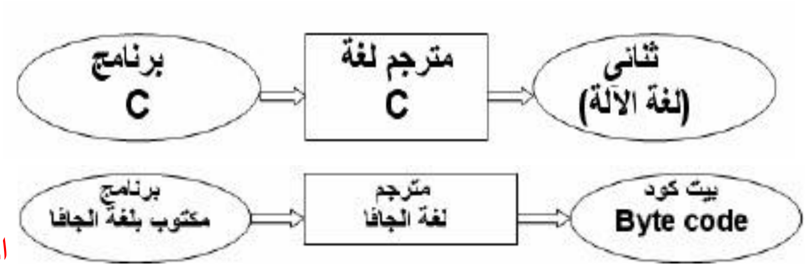
١- مشكلة حجز الذاكرة.
٢- التعقيد المفرط في لغة C++.
٣- زيادة مستوى الحماية و الأمان لاسيما في الشبكات و ذلك بتخلصها من مفهوم المؤشرات التي كانت سائدة في اللغات التقليدية.

ومن أهم مزاياها وخاصة ضمن مشروعنا :

١- ميزة البناء والتنفيذ التفرعي لأكوادها عن طريق الفكر الحديث للبرمجة من خلال مفهوم البرمجية الشيئية والرسومية OOP (Object Oriented Programming) وهي طريقة تعتمد على كتابة الكود البرمجي داخل مجموعة من الClasses حيث يحتوي كل Class على جزء من الكود المطلوب إدراجه ضمن البرنامج ويتم إدراج هذه الأكواد المبعثرة ضمن الClasses إلى داخل الدالة الرئيسية للكود main عن طريق أخذObject كائن منها .

٢- ميزة أنها اللغة التي تعتمدها لوحة IOIO/OTG لتتخاطب مع التطبيق المبني على أساس تعليماتها البرمجية لتتحكم بها مباشرة وكذلك لوحة الأردينو يغدو التخاطب مبسَّطاً أكثر بينها وبين لغة ArduinoC المتحكمة بلوحة Arduino UNO .

٣- البرنامج المكتوب بلغة الجافا يستطيع العمل على أي جهاز مهما اختلفت مواصفاته سواء hardware, software بعكس اللغات الأخرى كـ C ففي اللغات الأخرى يتم تحويل الكود البرمجي إلى لغة الآلة الصورة الثنائية Binary وهذه الصورة تعتمد كثيراً على نوع الآلة التي تمت البرمجة عليها وقد لا يصلح الكود للعمل على آلة أخرى . أما في جافا فعند مرحلة ترجمة الكود يتم ترجمته وتحويله مباشرة لصورة البايت ByteCode وهذه الصورة تصلح للعمل على أي جهاز مهما اختلفت مواصفاته أو نظام التشغيل .



الشكل(١.١)

ولو تحدثنا عن مزايا JAVA لغة الجافا لن نستطيع أن نحصيها نظراً لما وفرته من مزايا كثيرة أدت بدورها إلى الاستغناء عن اللغات الأخرى ، فما هذه إلا لمحة بسيطة عن JAVA لغة الجافا التي أدى ظهورها إلى إحداث ثورة كبيرة في عالم البرمجيات الجاهزة .

٣.١- موجز استخدام JAVA وكيفية البناء وكتابة الكود البرمجي :

هي لغة ضخمة جداً فبالنسبة للجزء من هذه اللغة والذي استخدمناه بمشروعنا كافة التعليمات الرئيسية البسيطة :

من ناحية الدوال الحسابية + - * / % والشروط المنطقية || && > < <= >=

والشروط & | وتعليمات التكرار واختبار الشروط (){} Switch

,If(){}else{},?:,while(){},for(){},do{}while(){},

وأنواع المتغيرات وتعريفاتها نفس طريقة لغة C++ من ناحية الاستخدام لكن من ناحية الموضع هناك طريقة يلتزم بها عند كتابة الكود ضمن الكلاس وذلك يكون بأنه ضمن كلاس الدالة الرئيسة يتم تعريف Classes ضمن كل كلاس الجزء الأول منه تعريف المتغيرات (المتحولات) ، الجزء الثاني عبارة عن دالة تدعى Constructor مشتقة من الكلاس فيها تعطى قيم للمتغيرات ، الجزء الثالث دوال معرفة نوع المتحولات التي ستخرجها بعد إجراء العمليات عليها المكتوبة ضمن هذه الدوال .

أما من ناحية أعقد:

فالتعليمات الجديدة ومفهوم الكائنات وبناء أكثر من كلاس ضمن آخر وتضمين مسارات Classesالمصدرية ومفهوم ال Activity ، Layout ، manifest و غيرها سنتطرق لها بإيجاز عند الحديث عن بناء تطبيقات الأندرويد ضمن منصات تطويره البرمجية وسنلحق أسماء مراجع إضافة لكتب تلحق بالمشروع لكي تكون سبيلاً ميسراً لتعلم هذه اللغة وتطوير هذا النوع من المشاريع في المستقبل لمن أحب التعمق بهذا المجال .

١.٤- اللغة البرمجية ArduinoC:

وهي مشابهة جداً للّغة البرمجية C++ من ناحية طريقة كتابة الأوامر وتنسيق الأكواد.

فهي تطوير اللغة البرمجية المستخدمة في لغة Processing

وتستخدم هذه اللغة ضمن بيئة مترجم Arduino IDE لكتابة الكود البرمجي المتحكم بمعالج لوحة الأردينو .

١.٥- أنظمة تشغيل الهواتف المحمولة (الأجهزة الذكية):

يمكن تصنيف أنظمة تشغيل الهاتف المحمول على أساس

الشركة المصنعة أو المطورة لنظام التشغيل

من أشهر أنظمة تشغيل الهواتف المحمولة نظام الويندوز من مايكروسوفت Windows Phone ، نظام سمبيون Symbian OS ، نظام بادا BADA ، نظام أي أو أس IOS ، الأندرويد Android و نظام بلاك بيري Black Berry .

أنظمة مفتوحة المصدر Open Source Operating System

هي أنظمة التشغيل التي يمكن الاطلاع والتعديل على شفرتها البرمجية وهى أكثر مرونة للمستخدم من الأنظمة الأخرى التي لا تتيح مرونة للمستخدم ، مثل نظام أندرويد .

أنظمة احتكارية Proprietary

تعد أنظمة التشغيل برمجيات احتكارية في عدة أحيان. عادة عندما يكون على البرنامج حقوق أو منشور تحت ترخيص EULA وعندما يقوم صاحب المشروع باستخدام طرق تقنية لمنع المستخدمين من التعديل على برنامجه مثل عدم نشر الكود المصدري للبرنامج. أو عندما يقوم هذا الشخص باستخدام طرق قانونية كحقوق النشر وتراخيص EULA لكي يمنع المستخدم من إعادة توزيع البرنامج على مستخدمين آخرين ، مثل جميع انظمة مايكروسوفت و أبل

١.٦- نظام أندرويد Android :

Android الأندرويد هو نظام تشغيل مفتوح المصدر للأجهزة المحمولة وليس فقط للهواتف المحمولة و يدعم تعدد التطبيقات ،أي ممكن أن يعمل به أكثر من تطبيق في نفس الوقت أو الخدمات في الخلفية له. وتعمل كل التطبيقات فيه بشكلٍ متساوٍ من حيث المشاركة في موارد الجهاز بالرغم من أن كل تطبيق يعمل بشكل مستقل بطبقة منفصلة . وتمكنت منصته من إعادة استخدام كائناته كما ويستخدم محرك (web Kit OSS) كمتصفح انترنت والذي يستخدم في متصفح (Google chrome) كما يدعم مكتبة (OpenGL ES) لإدارة الصور الثلاثية ويعتمد قواعد بيانات (SQLite) في تخزين البيانات . كما ويعم كل الصيغ القياسية لوسائل الميديا من صور و اصوات وفيديو والجميل فيه أنه يتضمن عتاد متنوع من الكاميرا الرقمية و نظام التموضع العالمي (GPS) والبوصلة بالإضافة الى دعم خاصية الإحساس بالمكان (Accelerometer) التي سنستخدمها ضمن المشروع .

إن تطبيقات الأندرويد مبرمجة بلغة الجافا وطالما أن لغة الجافا هي لغة محمولة portable وتستطيع أن تعمل على أي نظام تشغيل، فهي ميزة جيدة للإقبال على استخدامها ، بالإضافة الى أنَّ تطبيقات الأندرويد مجانية مما يؤدي إلى انتشارها بشكل أوسع بين الناس .

إذاً فهو نظام تشغيل مجاني ومفتوح المصدر Open Source مبني على نواة لينكس Linux صمم أساسًا للأجهزة ذات شاشات اللمس كالهواتف الذكية والحواسب اللوحية , يتم تطوير الأندرويد من قبل التحالف المفتوح للهواتف النقالة Open Handset Alliance الذي تديره شركة جوجل Google .

إذ في عام 2005 قامت Google بشراء Android الذي كان في هذا الوقت شركة مستقلة لا أحد يعرف عنها شيء ، و كان الإعلان الرسمي عنه في 5 نوفمبر 2007 بالتزامن مع انطلاق التحالف المفتوح للهواتف النقالة.أي في عام 2007 أصبح Android SDK متاحاً لتطوير

تطبيقات آندرويد وفى 2008 قامت HTC & T-Mobileبإصدار أول جهاز يعمل بنظام أندرويد باسم G1 وفي نفس العام ظهر Android SDK1.0 وأيضاً Android Market .

طرحت جوجل نظام أندرويد كنظام مفتوح المصدر تحت رخصة أباتشي Apache License ، مشروع أندرويد مفتوح المصدر (AOSP) هو الفريق المسؤول عن تطوير وتحديث وإصلاح النظام.

لدى الأندرويد مجتمع ضخم من المطورين الذين يقومون بكتابة وتطوير البرامج والتطبيقات للأندرويد، ويعتمدون بشكل أساسي على الكتابة بلغة الجافا JAVA .

شهد نظام الأندرويد عدداً من التحديثات منذ انطلاقه وأهمها :

خبز الزنجبيل (Gingerbread) وهو التحديث رقم 2.3 للأندرويد ، وفيه تم تحسين واجهة المستخدم ، لوحة المفاتيح لينة ، ميزة نسخ و لصق ، تحسين أداء الألعاب ، وأضاف دعم المكالمات عبر بروتوكول الإنترنت *Session Initiation Protocol* (*SIP*) .

قرص العسل (Honeycomb) وهو التحديث رقم 3.0 للأندرويد ، الذي يدعم الشاشات الكبيرة، ويقدم العديد من الميزات الجديدة في واجهة المستخدم ، ودعم المعالجات متعددة النواة وتسريع أجهزة الرسومات .

ساندويتش الآيس كريم (Ice Cream Sandwich) وهو التحديث رقم 4.0 للأندرويد ,وقد أُعلن عنه يوم 19 أكتوبر 2011، وفيه تم إضافة ميزات قرص العسل إلى الهواتف الذكية ، وإضافة الميزات الجديدة بما في ذلك التعرف على الوجه ، ورصد استخدام شبكة البيانات والتحكم ، الاتصالات الموحدة والشبكات الاجتماعية ، والتحسينات التصوير الفوتوغرافي .

جيلي بين (Jelly Bean) أندرويد 4.1\4.2 (جيلي بين) هو الإصدار الأخير للأندرويد الذي يتوفر على الهواتف ومن ابرز مميزاته سريعة وناعمة واكثر ما يميزه هو البحث الصوتي وخدمة Google Now التي تجلب لك المعلومة المناسبة في الوقت المناسب بالإضافة لتواجد كل مميزات الاصدار السابق ساندويتش الآيس كريم أو يمكن القول أنه فرع منه.

١.١ الإصدارات المختلفة:

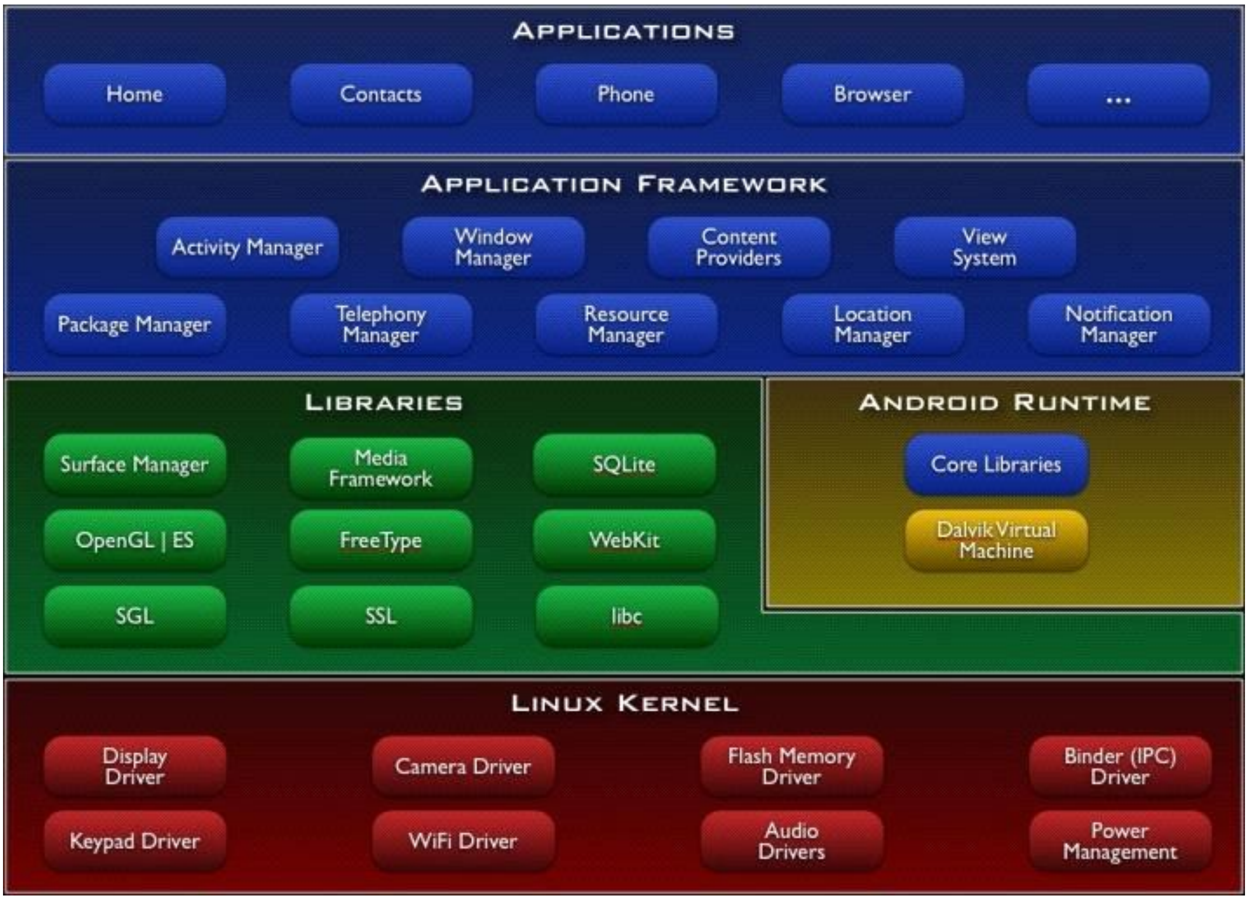
الإصدارات المتعارف عليها بالنسبة للمستخدمين العاديين تختلف عن الإصدارات التي يتحدث بها مطورو التطبيقات، حيث كل إصدار مختلف من أندرويد يقابله إصدار من الـ "API level" ،فإذا أردنا مثلاً أن نصنع تطبيق يستفيد من خصائص أندرويد فإن المستوى المقابل هو 11 منالـ "API Level"إن أندرويد لا يدعم بقوة لـ "Backward Compatibility" فإذا طورنا

تطبيق باستخدام إصدار معين منالـ "API Level" فإنه سيعمل على ما هو أحدث من هو لكن ليس بالضرورة أن يعمل على كل ما هو أقدم منه ،لذلك الاختيار هنا محيّر ويرجع إلى المطور نفسه وكيف يريد أن يسوّق تطبيقه.

Android versions x.x.4 ------- x.x.1

API levels 1 ------- 15

١.٢ البنية الداخلية للأندرويد:



الشكل (١ - ٢) البنية الداخلية للاندرويد

وتتألف من عدة طبقات:

١.٣ طبقة Linux Kernel :



الشكل(١ - ٣) طبقة *linux Kernal*

وهي مسؤولة عن:

- Display Driver
- Bluetooth Driver
- Camera Driver
- Flash Memory Driver
- Binder (IPC) Driver
- Keypad Driver
- USB Driver
- Wi-Fi Driver
- Audio Driver
- Power Management

تم بناء نظام الأندرويد في الأساس على النظام لينوكس لأن هذا النظام هو نظام مفتوح المصدر ولأنه يوفر خيارات كثيرة للتعامل والتخاطب مع الأجهزة . وإن بناء نواة جديدة سيكلف وقتاً وجهداً ومالاً لذلك تم تخصيص نواة لينوكس لتناسب أجهزة الهواتف الذكية .

١.٤ طبقة Libraries :



الشكل (١ - ٤) طبقة المكتبات

وهذه المكتبات هي:

- Surface Manager
- Media Framework
- SQLite
- OpenGL ES
- Free Type
- Web Kit
- SGL
- SSL
- Libbc

وهي أهم المكتبات التي يحتاجها المطور في بناء تطبيقاته. إن مهمة المكتبات تقديم خدمات للمطور لتسهل عليه تطوير التطبيقات.

المكتبة SQLite مثلاً ، يوجد قواعد بيانات جاهزة ومجانية ومفتوحة المصدر موجودة على النظام توضع في SQLite لنستفيد من خدماتها.

المكتبة Web Kit مثلاً ، تستخدم في المستعرضات وفي رسم شاشات المستعرض وصفحات الويب وتستخدم من أجل إمكانية استعراض المواقع من الأجهزة الذكية.

وفي كل مكتبة تحتوي توابع تدعم الاغراض التي تخدمها المكتبة.

١.٥ طبقة Android Runtime :



الشكل (١ - ٥) طبقة *runTime*

طبقة التشغيل هذه فيها:

- Dalvik Virtual Machine
- Core Libraries

مهمة هذه الطبقة تشغيل البرامج والتطبيقات وذلك بحجز مكان للتطبيق، أي أن كل برنامج يشتغل في الأندرويد تشغل معه نسخة خاصة من Dalvik Virtual Machine مما يعمل عزل كامل بين التطبيقات لكي لا تتداخل مع بعضها.

١.٦ طبقةApplication Framework :



الشكل (١ – ٦) طبقة *framework*

كل Framework مقسم الى Manager يقدم خدمات معينة يستفيد منها المبرمج في بناء تطبيقاته مثلWindow Manager يختص في بناء الواجهات والشاشات من حيث الشكل والحجم وغيرها ويقدمها لنا بحيث لا ندري كيف ترسم هذه الشاشة .وبالتالي يستفيد المبرمج من هذه الطبقة في بناء برامجه.

١.٧ طبقة Application :



الشكل (١ ـ ٧) طبقة التطبيقات

في هذه الطبقة الاخيرة تأتي التطبيقات الرئيسية التي تضعها الشركة مثل:

- Home
- Contacts
- Phone
- Browser

هذه البرامج تتصل مع الطبقات التي تحتها عن طريق طبقة Application Framework .

١.٨ الوحدات الأساسية المستخدمة في بناء أي تطبيق أندرويد :

١.٨.١ Activity

Activityهي وحدة البناء الرئيسية لأي تطبيق، ويمكن اعتبار أي Screen داخل التطبيق عبارة عن Activity .

لكي نقوم بتنظيم محتويات الـ Activity على الشاشة نستخدم Layouts وهي عبارة عن ملفات XML يكون بداخلها التوصيف لهذه المحتويات وكيفية تنظيمها ويمكن أن يكون للـ Activityالواحدة أكثر من Layout .

أي أن Activity تستخدم لتوصيف الشاشة ، وهي عبارة عن class يتم استدعاؤها عن طريق الزنادIntent . و يمكن جعلها تعمل في الخلفية أو إعادة استدعائها أو إغلاقها نهائياً كما هو موضح في الشكل ()الذي يمثل دورةالحياة Activity Lifecycle

فكود الجافا كود الActivity لتوصيف الشاشة يكتب ضمن مجلد src ويكون امتداد الكود .java

أما كود الواجهة الرسومية Layout فيكتب ضمن ملف res ويكون امتداد الكود .xml

ثم يتم الربط بين الكود والواجهة في مجلد مرجعي R.java ويتم توليد التطبيق بصيغة APK ضمن ملف bin بعد تحديد نوع الإصدار الهدف والسماحيات المتاحة له بمجلد السماحيات mainfest

النشاط Activity Launched يتعامل مع الأحداث:

onCreate(): يتنفذ الحدث عند الإنشاء ،نضع فيه عادة التهيئة الأولية.

onStart(): يتنفذ الحدث عندما يصبح النشاط مرئي (الشاشة مرئية) .

onResume(): يتنفذ الحدث عند استئناف عمل النشاط، والمستخدم هنا يتفاعل مع الشاشة من جديد.

عندما يصبح النشاط في حالة Running أي أن النشاط فعال ولايوجد حدث، في هذه الحالة يتعامل مع الأحداث:

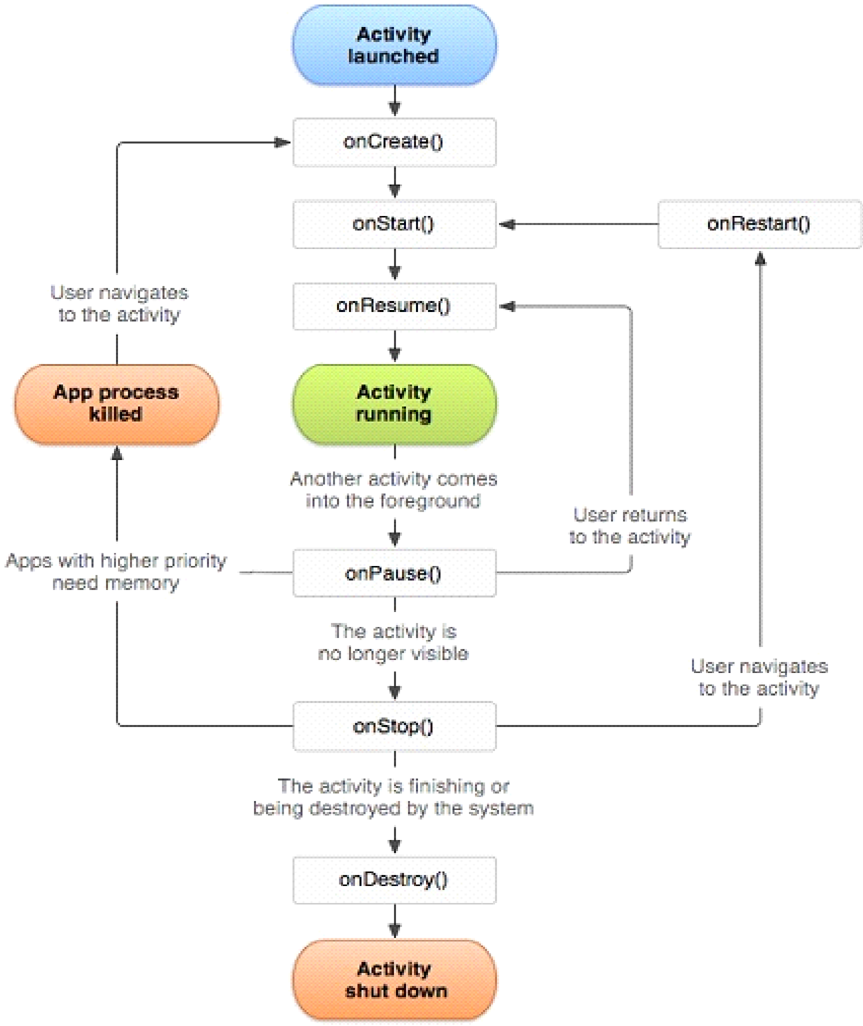
onPause(): يتنفذ الحدث عند توقف النشاط، أي عند طلب شاشة ثانية. أي في الفترة الانتقالية من الشاشة الأولى إلى الشاشة الثانية.

في هذا الجزء من دورة الحياة قد يحتاج الأندرويد إلى ذاكرة إضافية في حال وجود العديد من الشاشات في وضعية onStop() , onPause() والتي تبقى موجودة في الذاكرة بدون فائدة لذلك يقوم بإغلاقها بواسطة عملية kill() وعندما يحتاجها مرةأخرى يقوم بعملية onCreate() .

onStop() : يتنفذ الحدث عند اختفاء النشاط، أي عندما تختفي أول شاشة لتظهر الشاشة الثانية بدلاً منها.في هذه الحالة البرنامج يبقى موجود في الذاكرة.

onRestart(): هنا تعود الشاشة الأولى لتظهر أمام الشاشة الثانية وتنفيذ الحدث onStart() وإكمال عملها وفق دورة الحياة.

onDestroy(): هي عملية إغلاق نهائية للنشاط ،مثل عملية kill() والتي تنهي النشاط وتحذفه من الذاكرة.



الشكل (١ - ٨) مخطط حياة ال *activity*

١.٨.٢ Intent :

يعتبر الزناد Intent من الأشياء التي يتميز الأندرويد بشدة عن باقي أنظمة التشغيل، حيث أنه يعطي للمطور إمكانيات كبيرة للتحكم الكامل بالتطبيق ويمكّنه أيضاً من التعامل مع التطبيقات الأخرى المحملة على الجهاز .فهو يُستخدم لاستدعاء نشاط آخر في التطبيق .ويمكن اعتباره على أنه رسالة محملة بمعلومات معينة تنتقل من نشاط إلى نشاط آخر.أو يمكن أن يستخدم لاستدعاء أيقونة أو أي رابط معين لصفحة انترنت. ...

قد يقوم الزناد بفتح نشاط ثانٍ وإرسال بيانات له، أو قد يقوم بفتح نشاط وانتظار بيانات منهل يعود بها إلى النشاط الأول.أي أن الـ Intent يخدمنا في الـ Actionالذي نحدده ليقوم به ، وفي الـ Dataالتي يرسلها للنشاط الآخر أو يستقبلها منه.والـ Data التي نضعها فيالـ Intent ترسل في حزمة Bundle على شكل جدول مقسوم إلى قسمين key ، value لكل معلومة.

١.٨.٣ Background Service :

هي خدمات تعمل في الخلفية ولا يوجد لها واجهة رسومية مثل (العمليات الخاصة باتصال الانترنت، تشغيل موسيقى في الخلفية...).

وهي تعمل باستقلالية عن العنصر الذي تم تشغيلها من خلاله ،فمثلاً إذا تم تشغيلها عن طريق Activityمعينة ثم تم إغلاق هذه الـ Activity لاحقاً فإنها تستمر في العمل ويجب إيقافها يدويا.

يمكن أيضا أن تكون الـ Service متاحة لأي عنصر من عناصر التطبيق أو التطبيقات الأخرى إذاسمح المطور بذلك.

١.٨.٤ ملفAndroid Manifest:

أحد أهم الملفات في أي تطبيق أندرويد، ملف تحديد السماحيات وهوعبارةعن ملف xml يحتوي على توصيف لكل شيء يخص التطبيق (مثل إصدارالـ API Level المستخدم في تطويرالتطبيق،أنواع السماحيات Permissions التي يحتاجها التطبيق، توصيف للـActivities ..).

يمكن تعديل هذا الملف عن طريق واجهة رسومية أوعن طريق تعديل الكود مباشرة لكن الكود يعطي تحكم أكبر من الواجهة. ويتم إنشاء هذا الملف تلقائيا عند إنشاء أي مشروع.

١.٨.٥ Broadcast Receivers :

وهي المسؤولة عن استقبال كل الأحداث القادمة من الـ System مثل (عندما تطفأ شاشة الجهاز، عندما تكون البطارية منخفضة عندما يكون اتصال الانترنت متاحا،...)

وهي أيضا تعمل في الخلفية ولا يوجد لها واجهة رسومية لكنها من الممكن أن تظهر تنبيهات للمستخدم (Status bar notifications) لإعلامه بحدوث شيء ما.

يمكن أن ينشئ التطبيق Broadcasts خاصة به ويمكن للتطبيقات الأخرى استقبالها.

١.٨.٦ Content Providers:

تستخدم لحفظ واسترجاع الـ Data وجعلها متاحة لكافة التطبيقات، وذلك عن طريق إعطاء Public URI لكل Data Set يتم التعامل معها عن طريقه. وهي الطريقة الوحيدة لمشاركة البيانات بين التطبيقات.

يحتوي الأندرويد على العديد من الـ Content Providers مثل(... ,Contacts, Media) .

١.٩ SDK:

هي حزمة المكتبات التطويرية البرمجية التي تحتوي المجموعات والدوال التي يمكن استخدامها في تطوير تطبيقات الأندرويد ، وتضم بيئة تطوير وبرمجة داخلية (IDE) ، تحتوي بذاتها واجهة لكتابة الكود المصدرية (Source Code) ومتتبع أخطاء ومصمم رسومي لتعديل واجهة المستخدم الرسومية (GUI) وبالتأكيد فإنها تضم مترجما (Compiler) لإخراج الصيغة القابلة للتنفيذ من التطبيق .

١.١٠ ADT:

هي عبارة عن مجموعة أدوات خاصة ببرمجة الأندرويد فهي تزودنا بالواجهات الرسومية والأدوات (مثل الأزرار ومربعات النصوص .... الخ) .

١.١١ AVD :

هو عتاد الأندرويد الافتراضي ويستخدم من أجل تشغيل محاكي هاتف الأندرويد Emulator، ويجب إنشاء هذا العتاد الذي من خلاله نحدد الإصدار من أندرويد الذي سنستعمله وبشكل عام يسمح لنا AVD بعمل تعديلات على Emulator لكي يعمل حسب رغبتنا ويدعم نوع الهاتف الذي نملكه والذي يجب أن يدعم نظام تشغيل الأندرويد .

١.١٢ APK:

عند إنشاء تطبيق للأندرويد اسمه app مثلاً وعندما نقوم بتشغيله run فإنه ينتج لدينا ملف بامتداد apk ويكون هذا الملف app.apk هو البرنامج الذي سيتم تنزيله على الأجهزة المحمولة .

١.١٣ Eclipse :

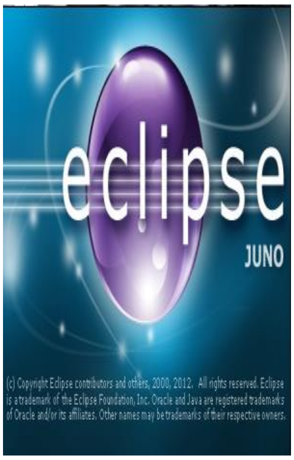
Eclipse هو عبارة عن هو بيئة متكاملة لتطوير برمجيات IDE مكتوب بلغة الجافا JAVA وهو مجاني ويخضع لترخيص مفتوح المصدر الذي يخضع له كذلك نظام تشغيل الأندرويد Android بالإضافة إلى لغة الجافا JAVA ، يدعم Eclipse لغات البرمجة الأخرى Python، C# ، Pascal و غيرها .

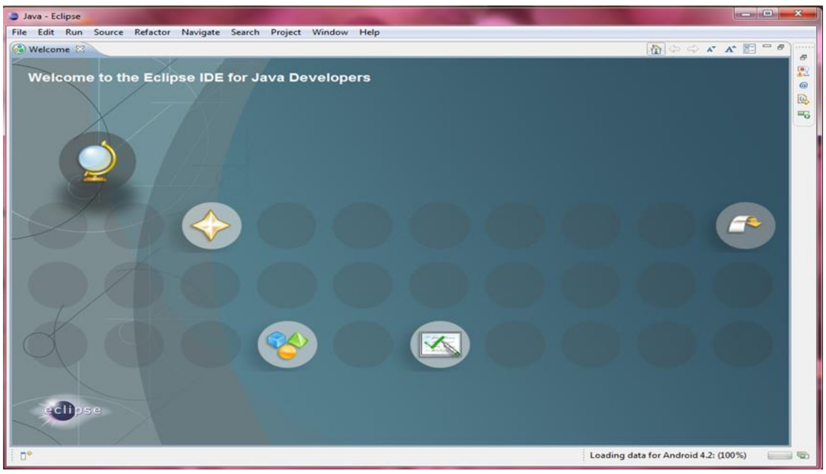
برنامج IDE أي يقوم بتسهيل عملية بناء التطبيقات فهو يوفر لنا عدة أدوات للتعامل معها مثل محرر الكود والمترجم ومصحح الاخطاء بدلاً من التعامل معها عن طريق سطرالأوامر.

هذه البيئة تتعامل مع لغة الجافا وتعتمد على مبدأ الإضافاتplugin فمن أجل التعامل مع الأندرويد ومن أجل توفير البيئة المناسبة لكتابة برامج الأندرويد نقوم باستخدام Eclipse ونضيف إليه الأدوات التي تسمح لنا بتطوير تطبيقات على الأندرويد مثل SDK ، AVD، ADT .

رغم أن Eclipse مجاني و مفتوح المصدر إلا أنه غني بالوثائق التي تساعد على تطوير برمجيات و حل المشاكل التي قد يواجهها المبرمج ، و من الجدير بالذكر هنا بأن Eclipse يعمل بنفس الكفاءة على مختلف الأجهزة و أنظمة التشغيل المختلفة وهذه ميزة لها قيمتها .

و بالعودة لتاريخ Eclipse فهذه البيئة تم تطويرها من قبل شركة IBM ابتداءا من عام 1996 و الهدف كان تطوير تطبيقات خاصة مكتوبة بلغة جافا ولحسن الحظ ابتداءا من عام 2004 تم طرح بيئة Eclipse كبرنامج مجاني مفتوح المصدر مما سمح للجميع بتطويره و عمل تعديلات عليه دون قيود . ونشير هنا إلى أن Eclipse يتعامل مع الإضافات Plug-in بشكل واسع حيث يمكن برمجة الإضافات و دمجها في بيئة Eclipse ومن ثم الشروع في برمجة التطبيقات المطلوبة . طبعاً هناك برنامج آخر أفضل حالياً اسمه Android studio

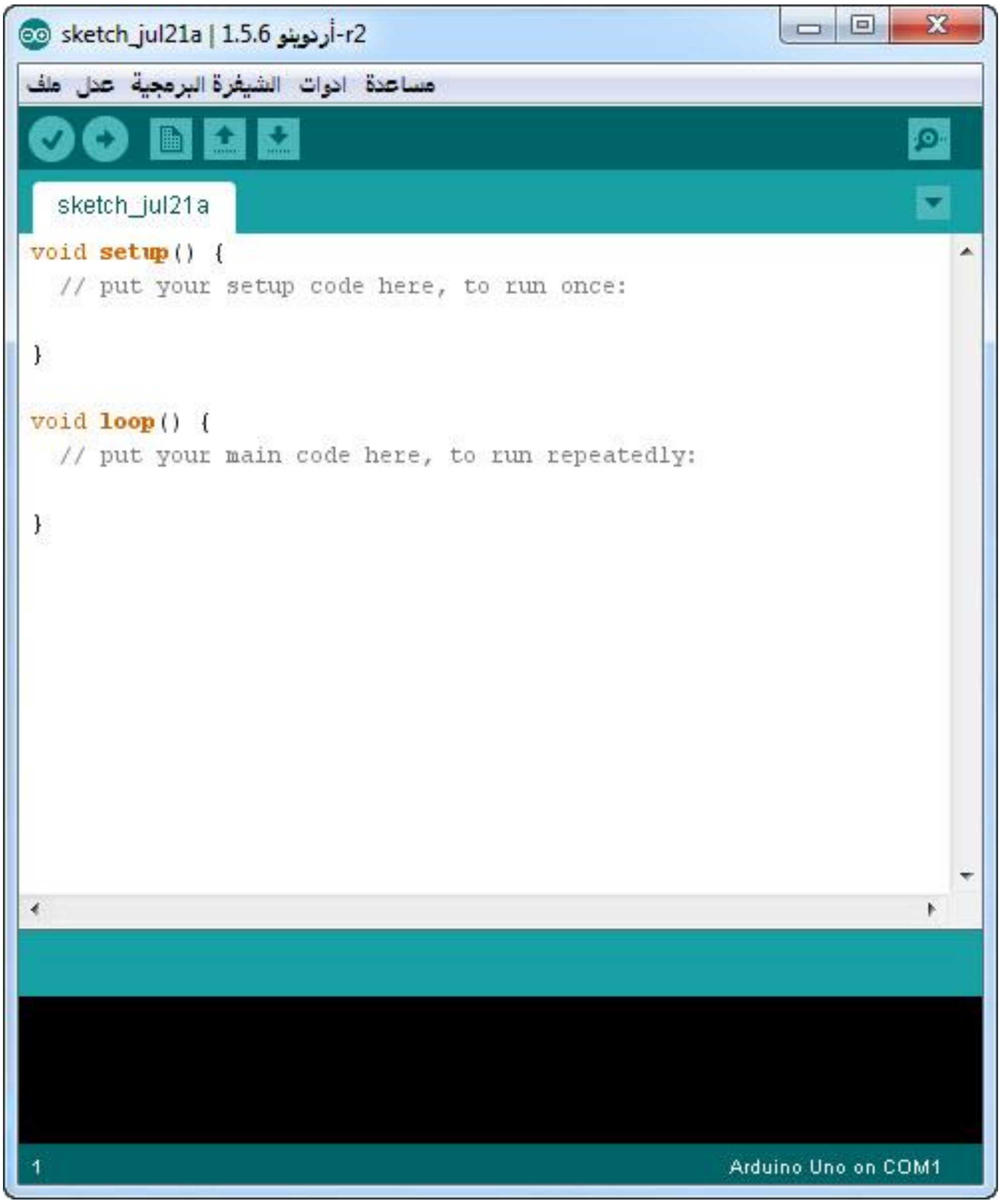




بالتالي روابط مهمة لتحميل البرامج هي:

1. Android SDK (Android Studio, no longer Eclipse)
http://developer.android.com/sdk/index.html
2. All the Android tools and Android APIs (e.g. Android 4.4.2…). This might take a couple of hours
Some packages might not get installed but that’s ok.
http://developer.android.com/sdk/installing/adding-packages.html
3. The latest IOIO libraries (Client Software, 5.06/5.05: App-IOIO0505.zip)
https://github.com/ytai/ioio/wiki/Downloads

١.١٤ Arduino IDE :



برنامج يتم فيه تعريف المكاتب البرمجية للملحقات العتادية لبورد الأردينو وفيه يتم كتابة كود التهيئة للشريحة و الأكواد التكرارية بلغة ArduinoC . http://www.arduino.cc/

١.١٥ Genymotion المحاكي :

عبارة عن جهاز موبايل افتراضي يحمل نسخة نظام تشغيل أندرويد تخولنا تشغيل التطبيقات المبرمجة بدون الحاجة لجهاز أندرويد حقيقي لأنه يسمح باختبار الAPK المولَّد ورؤية مدى صحة الإضافات والتعديلات من ضمن شاشة الحاسب نفسه .



يمكننا القول أنه في النهاية متطلبات البرمجة هي : معرفة لغة البرمجة C وأساسيات لغة الجافا العامة والجافا الغرافيكس وتعليمات اليويو بورد بالجافا .

٧.١- نظرة عامة على أساسيات مكتبة IOIO :

تعرض المكتبات أماكن واجهات الجافا ، و توضح المزايا المختلفة للشريحة عندما تقوم ببناء تطبيقك الخاص.

مكتبات الــ IOIO تحتوي على مجموعة من لواحق الملفات jar أو apk لذلك تطبيقك سيكون مكتف ذاتياً و لن يحتاج إلى الاعتماد على مكونات ميزة التنصيب . الآن لنشرح ما هي المكتبات الموجودة وما الهدف من كل منها .

IOIOLib هي مجموعة من المكتبات ، من أجل الأندرويد و من أجل الحاسب ، التي تتيح لتطبيقك ان يتحكم بشريحة IOIO .

مكتبة IOIOLibPC (تعمل في بيئة Eclipse والتي تندرج تحت IOIOLib/target/pc) و هي تكون المكتبة المراد استخدامها لتفعيل IOIO من الحاسب .

مكتبة IOIOLibAndroid (تعمل في بيئة Eclipse والتي تندرج تحت IOIOLib/target/android) وتكون المكتبة المراد استخدامها لتفعيل IOIO من جهاز الأندرويد ، هناك مكتبتان إضافيتان IOIOLibBT , و IOIOLibAccessory وتكونان تكملة لمكتبة الــ IOIOLibAndroid و تضيف خاصية Bluetooth ووظائف Android Open Accessory كتمكين للاتصال مع الــ IOIO ، بالتتابع .

السبب وراء الفصل لمكتبة IOIOLibAndroid التي تعمل في أي إصدار من إصدارات أندرويد (كالإصدارات الأخيرة كــv1.5)عنهما أن الــBluetooth الموجود في الأندرويد v2.x مع الإكسسوارات (المصادر) المفتوحة فقط متوفرة في اختيار نظام الأندرويد 2.3.4 أو الأجهزة الأحدث . اذا تم ربط أي مكتبة أو أكثر إلى تطبيقك المستخدم ، سوف ينتشر اتصال الـIOIO بشكل أتوماتيكي عبر أي من القنوات المتوفرة كقناة USB، شرط أن يكون التطبيق يعتمد على أدوات الأندرويد المخصصة .

مكتبة IOIOLib العامة الكاملة API تملك وثائق JAVA قياسية ، التي تسعى لتكون شاملة و تستخدم كمرجع عند كتابة الكود .

تأسيس واستخدام IOIOLib

يتم تعريف IOIOLib لمكان ما حيث يتم حفظ المشاريع بشكل طبيعي Workspace ويتم استيراد المكتبات المناسبة المستخدمة في Eclipse عن طريق :

File > Import... > General > Existing Projects>workspace

ومن ثم نختار المكتبة IOIOLib وبالتالي تكون المكتبة أنشأت طبعاً مع التضمين لها في ال properties .

وعند تطوير تطبيق الأندرويد تأكد من IOIOLibAndroid و اضغط "Finish " .

ولإضافة مكتبة Bluetooth و ، أو موصلات Android Open Accessory ، نكرر العملية من أجل IOIOLibAndroid و مكتبات IOIOLibAccessory. علينا التأكد أن التطبيق يقوم باستخدام سماحية android.permission.INTERNET وسماحية android.permission.bleutooth حيث يمكن اضافتها بواسطة فتح الملف الموجود على جذر المشروع AndroidManifest.xml ، والذهاب إلى السماحيات عن طريق

tab > Add... > Uses Permission > android.permission.INTERNET

علينا التأكد من تصحيح الأخطاء على جهاز الأندرويد من خلال الذهاب عبر

Settings > Applications > Development > Enable USB Debugging.

عند تطوير تطبيق على الحاسب تأكد من IOIOLibPC و اضغط "Finish" . أشر الى IOIOLibPC على مشروع التطبيق ، بواسطة تعديل بناء إعدادات تطبيق JAVA.

IOIOLib تتضمن قسمان رئيسيان هما

The IOIOLib Core API والذي يتألف من وسيلة مستقلة API من اجل التحكم بكل وظائف IOIO

The IOIOLib Application Framework والذي يتألف من وسيلة تطبيق ذو هدف خاص لتسهيل إنشاء عدة أنواع مختلفة من التطبيقات . فليس ضروريا استخدام بنية التطبيقات من أجل العمل مع شريحة IOIO ، لكن ينصح باستخدامها بشكل كبير ، منذ ظهورها هناك عدة عمليات ذاتية التشغيل قد اختفت و بدأ ظهور نظام التمرير الذي أصبح مطلوباً من اجل تطبيقات IOIO القوية و المرنة . إن المكتبات في JAVA منظمة في عدة مجموعات .

ioio.lib.api مجموعة تحتوي على IOIOLib الذي يعتبر الجزء الاساسي لـ API من أجل التحكم في IOIO ، هذه المجموعة سوف تستخدم في كل التطبيقات ، حيث تقوم بعملية المطابقة بين مكتبات أندرويد و مكتبات الـ PC ، لذا فإنه من السهل إنشاء التطبيق على PC للعمل على بيئة Android و العكس صحيح لذا فليس هناك حاجة للعمل على واجهتين مختلفتين .

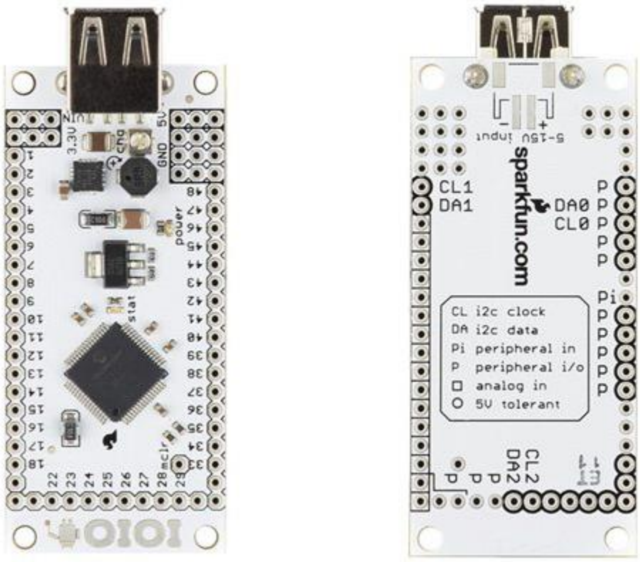
تحتوي مكتبة ioio.lib.util على تطبيق IOIOLib الذي يسهل العمل على تطبيقات IOIO عامة . تتضمن هذه المكتبة أيضا مكتبات فرعية خاصة مثل ioio.lib.util.android التي تسهل

العمل على بيئة Android التي تسهل عمل التطبيقات على شريحة IOIO ، كما تساعد مكتبة ioio.lib.util.pc على إنشاء تطبيقات JAVA المتنوعة على PC .

بالنسبة لاستخدام اليويو بورد وتعليماتها وتطبيق ذلك على منافذها سيتم شرح ذلك في الفصل الثاني في قسم التعريف بشريحة اليويو ضمن المكونات الصلبة .

الفصل الثاني

المكونات الصلبة



الشكل (1)

١.١ شريحة IOIO ولماذا ؟

صممت شريحة IOIO بشكل خاص للعمل مع بيئة أندرويد الشكل(1) ,حيث أنها تقدم اتصالاً قوياً مع بيئة أندرويد عبر USB او Bluetooth وقابلة للتحكم بشكل كامل من قبل تطبيقات أندرويد باستخدام تطبيقات JAVA البسيطة ومصمم الشريحة

العالم اليهودي الأميركي **Ytai Ben-Tsvi** صممها في **30 Apr 2011**. بناء على الربط بين التعريف الفيزيائي لبنيتها لتعريف برمجي عن طريق بيئة Pspice ثم ربط تعليمات إنشائها في هذه البيئة بــClasses برمجية بلغة الجافا وبالتالي إنشاء مكاتب مرجعية لها وبذلك أصبح تستطيع التخاطب مباشرة مع تطبيقات الجافا بعد أن برمج لها Bootloader firmware .

تحتوي شريحة IOIO على معالج مصغر يقوم بالاتصال عبر كابل USB واستقبال الأوامر من تطبيق أندرويد بالإضافة الى ذلك تستطيع هذه الشريحة التعامل مع طرفيات الأجهزة بنفس الطريقة مثل المداخل والمخارج الرقمية ، PWM، المداخل التشابهية ، بروتوكول الاتصال I^2C , بروتوكول البرمجة التسلسلية SPI ، التحكم ببروتوكول الاتصال التسلسلي UART ، جميعها يمكن استخدامها في شريحة IOIO مثل المعالجات الأخرى .

تتم كتابة النص البرمجي المستخدم للتحكم بهذه الواجهات بنفس الطريقة التي تتم بها كتابة تطبيق أندرويد وذلك بمساعدة مكتبات خاصة ، بمعنى آخر نستطيع استخدام قوة المعالجة ، الانترنت ، Bluetooth ، شاشة اللمس ، الحساسات المتنوعة المدمجة في جهاز أندرويد مع القدرة على إضافة الطرفيات بسهولة للتعامل مع العالم الخارجي . بالإضافة الى ذلك لا يحتاج

استخدام شريحة IOIO إلى أي معدلات هاردوير او سوفت وير للارتباط بجهاز أندرويد ، كل ذلك يأتي مع ضمان قدرة التشغيل بشكل صحيح بعيدا عن خطر الاختراق .

إن شريحة IOIO تعمل بنمط HOST اثناء اتصالها عبر USB مع أغلب اجهزة الأندرويد التي لها القدرة على الاتصال بنمط Slave . إن شريحة IOIO تحتوي على برنامج تشغيل ذو الإصدار V3.04 لذا فهي قادرة على التعامل مع احدث التطبيقات و هذه الشريحة قابلة للتطوير وتحظى بدعم واسع .

تحتوي هذه الشريحة على

١- **متحكم مصغر من عائلة PIC** ذو مزايا متطورة و هو مدعوم بمنفذ اتصال تسلسلي عام USB يعمل بنمط HOST و يحتوي على العديد من المداخل و المخارج الرقمية و التشابهية التي يتم التحكم بها عن طريق الأوامر القادمة من الكومبيوتر اللوحي .يحتوي أيضا على 64Pin ، 6Port

٢- **منظم جهد تقطيعي TPS62112** ذو تيار عالي يصل حتى 1.8 أمبير، دخله يتراوح بين (5-15) فولت يؤمن خرج قدره 5 فولت يستخدم لتغذية منفذ VBUSS USB (Pin 34) و يستخدم لتغذية منظم الجهد IC1 الذي خرجه 3.3 فولت

٣- **منظم جهد IC1** يؤمن خرج قدره 3.3 فولت ثابت و يستخدم لتغذية المعالج PIC24F

٤- **مقاومة متغيرة R2** (Trimpot smd) تستخدم لملائمة منافذ USB للكومبيوتر اللوحي و الشريحة

١.٢ مواصفات المعالج PIC24FJ256DA210:

وحدة المعالجة المركزية ذات الأداء العالي

- بنية هارفرد المحسنة
- ١٦ مليون تعليمة بالثانية عند تردد 32Mhz
- مذبذب داخلي بتردد قدره 8Mhz
- ضارب هاردوير بعرض 17Bit × 17Bit خلال نبضة ساعة واحدة
- مقسم هاردوير بعرض 16Bit\32Bit خلال نبضة ساعة واحدة
- مسجل مصفوفة فعلي بحجم 16×16 بت
- مترجم لغة C معدل مؤسس بنظام عنونة مرن
- ذاكرة برنامج خطي العنونة حتى 16Mbyte
- وحدتي عنونة منفصلة لكل من القراءة و الكتابة في الذاكرة

إدارة الطاقة

- منظم جهد مثبت على الشريحة بقيمة 1.8 فولت
- التنقل بين مصادر نبضات الساعة خلال زمن حقيقي

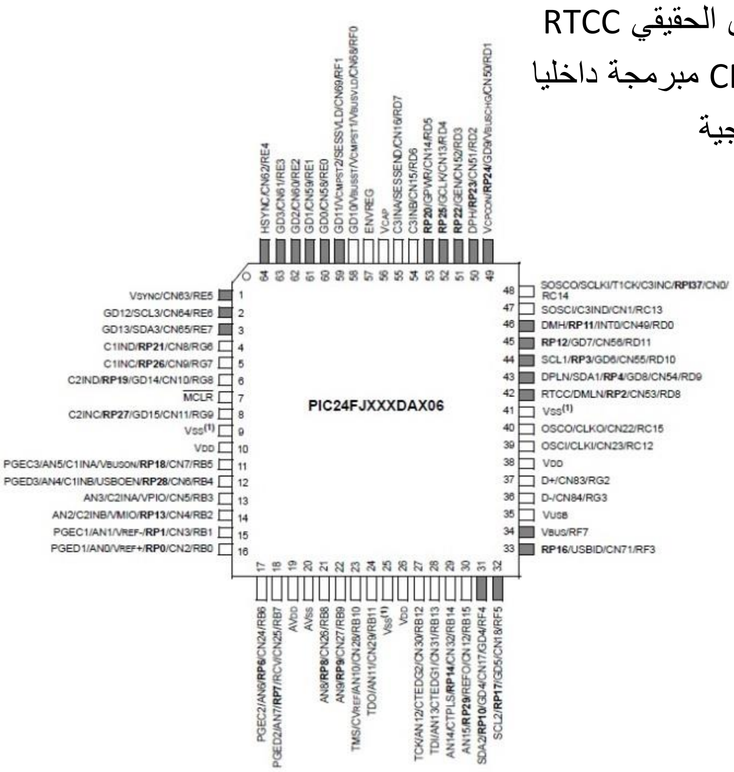
- حالة العمل : 800uA/MIPS ، 3.3v
- في حالة الراحة ينخفض التيار إلى 20uA
- حالة الاستعداد عند تردد مذبذب قدره 32Khz ينخفض التيار إلى 22uA

ميزات المتحكم الخاصة

- جهد التشغيل يتراوح بين 2.2v-3.6v
- مداخل رقمية بجهد قدره 5.5v
- مداخل و مخارج رقمية قابلة للتعديل من نوع مصرف مفتوح
- تيار مرتفع يصل إلى 18mA على جميع منافذ الدخل و الخرج
- وضعيات ادارة الطاقة قابلة للاختيار
- تصفير الي عند التغذية POR
- تصفير عند انخفاض التغذية BOR
- مؤقت مراقبة ذو هزاز RC داخلي مرن لتشغيل موثوق
- نافذة تسلسلية ICSP لتحميل برنامج المعالج
- ذاكرة برمجة وميضية

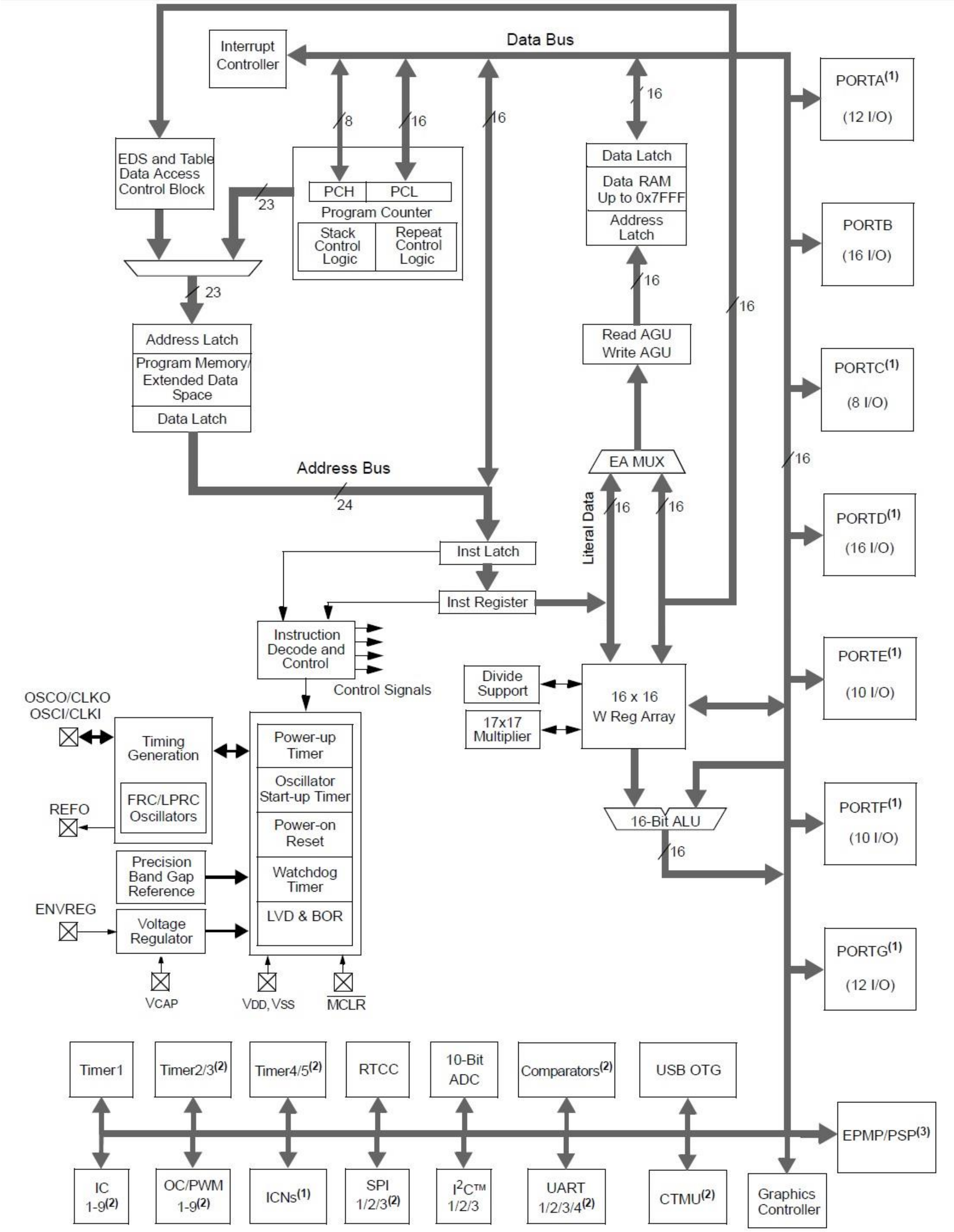
ميزات الطرفيات

- منافذ دخل\خرج تفرعية محسنة (EPMP/PSP)
- أربع نماذج UART
- خمس مؤقتات \ عدادات بعرض 16Bit مع مقسم مبرمج
- تسع عدادات بطول 16Bit ، لكل منها مقسم نبضات ساعة خاص
- تسع مخارج PWM لكل منها زمنه الخاص
- ساعة و تاريخ بالزمن الحقيقي RTCC
- مولد شيفرة تحقق CRC مبرمجة داخليا
- خمس مقاطعات خارجية



الشكل (2)

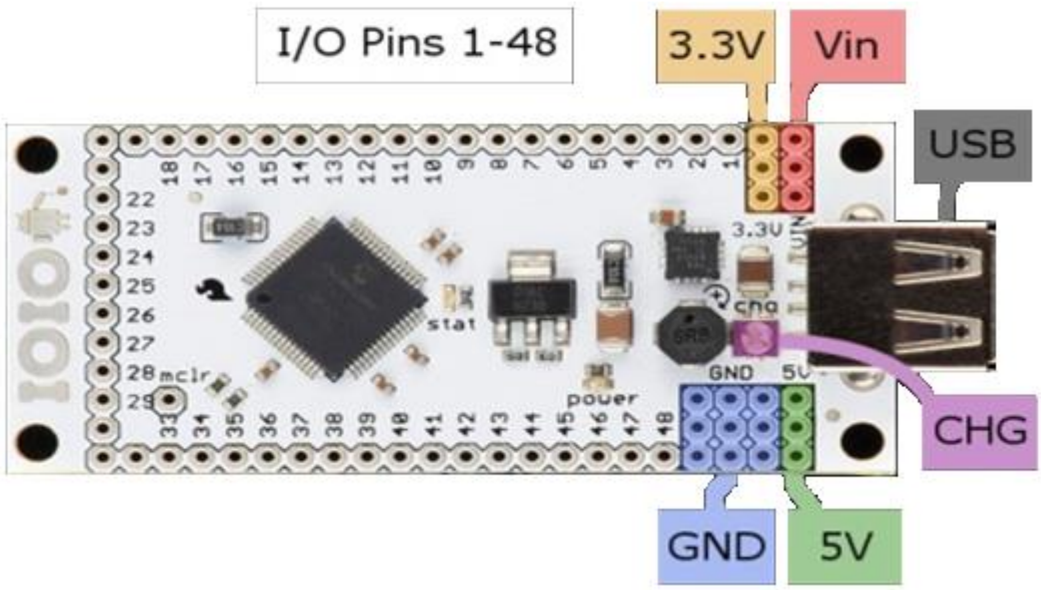
المخطط الصندوقي



الشكل(3)

١.٣ التعرف على شريحة IOIO:

إن شريحة IOIO تحتوي على المكونات التالية :



الشكل (4)

منفذ USB (النموذج A) مدخل يستخدم لوصل الشريحة مع جهاز الأندرويد

منافذ GND (9 منافذ) وصلات الأرضي .

منافذ Vin (3 منافذ) يستخدم لتغذية الشريحة الجهد المستخدم الواجب تطبيقه يتراوح بين (5-15) فولت.

منافذ 3.3 V (3 منافذ) : مخرج 3.3 V

منافذ 5V (3منافذ) يستخدم بشكل طبيعي كمخرج 5V عندما تكون الشريحة مغذاة عن طريق منفذ Vin

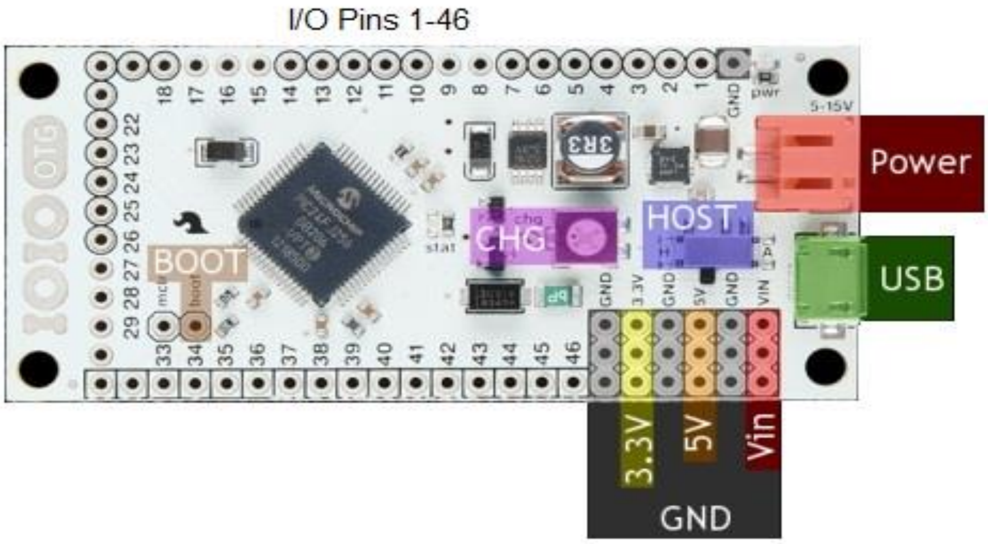
منافذ I/O (48 منفذ ، مرقمة من 1-48) الهدف الأساسي من منافذ I/O بعضها لديه وظائف خاصة ، تتم مناقشته لاحقا

ليد التغذية يضيئ عند تغذية الشريحة بالجهد .

ليد الحالة يضيء لمدة قصيرة عند إقلاع الشريحة .وبعدها يصبح تحت سيطرة البرنامج المطبق .

منفذ mclr ليس مستخدما بالحالة الطبيعية ، الغاية منه من أجل برمجة firmware Bootloader جديدة على شريحة IOIO في حال حدوث خطأ ومحوها من ذاكرة المعالج .

مقاومة متغيرة مسؤولة عن تحديد تيار الشحن (CHG) تضبط كمية التيار المطبقة على خط VBUS الموجود داخل منفذ الـ USB المتصل مع جهاز الأندرويد . بالتحريك بالاتجاه الموجب (+) يزداد تيار الشحن .

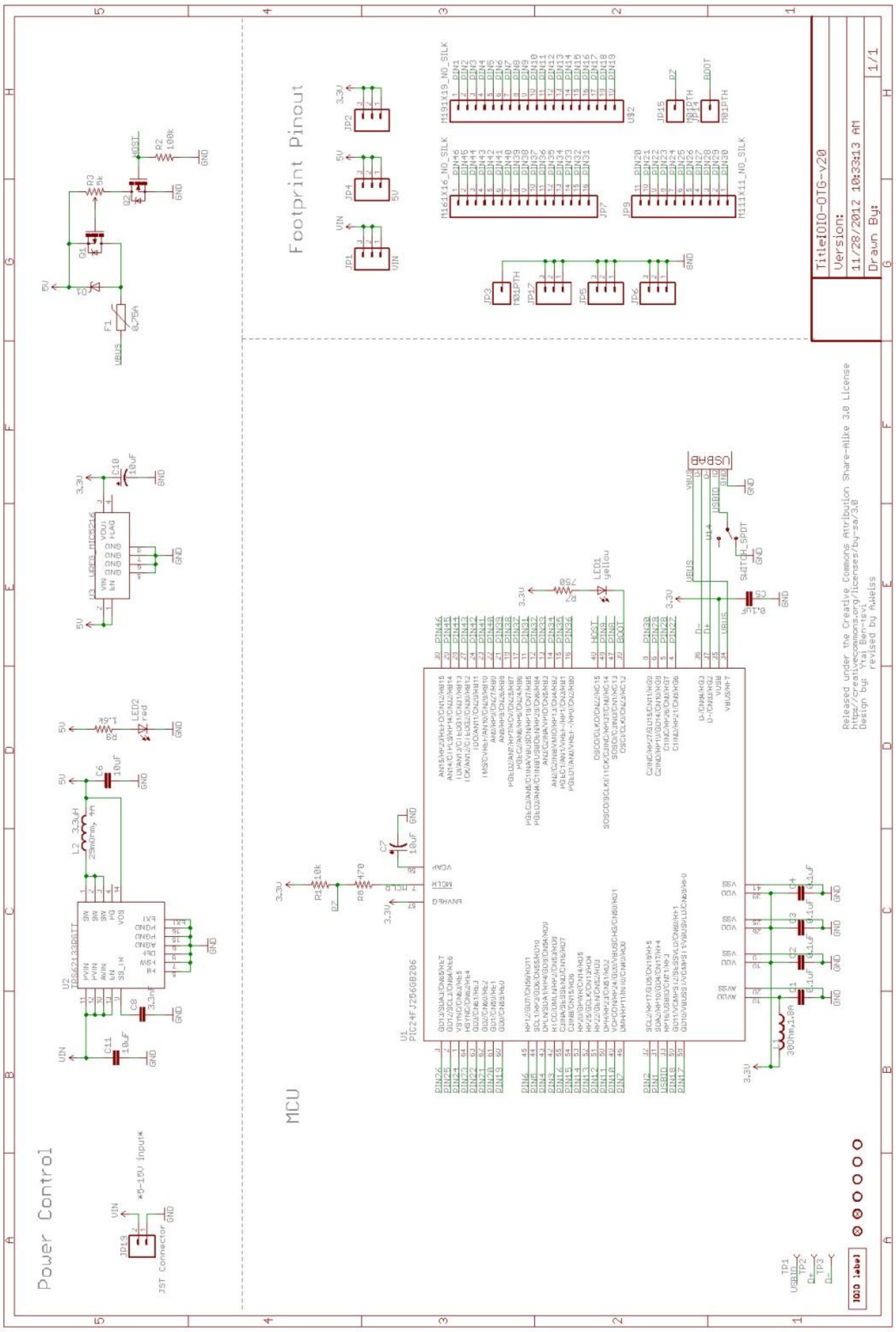


الشكل(5)

١.٤ منافذI/O في IOIO :

إن منافذ I/O هي جوهر شريحة IOIO ويتم استخدامها لوصلها بدارات خارجية أخرى مستخدمةً واجهات مختلفة ، القاسم المشترك لجميع المنافذ هو أنها يمكن أن تعمل كمداخل رقمية 3.3v او مخارج رقمية 3.3v ، بالإضافة إلى ذلك فإن العديد من المنافذ تكون قادرة على القيام بوظائف أخرى ، كما هو موضح في الجدول في أسفل هذه الصفحة .

من أجل الملائمة ، وظائف المنفذ المشترك ممثلة بيانياً على أسفل الشريحة، مع Legend ، المنافذ محاطة بمربع يمكن أن يستخدم كمداخل تشابهية (3.3v)، المنافذ محاطة بدائرة تكون ذو لها مجال العمل 5v ، مثال يمكن أن تستخدم كمداخل منطقية أو تنتج خرج 5v منطقي عندما تستخدم في نمط العمل المصرف المفتوح بمساعدة مقاومة الدفع . لتفاصيل أكثر أنظر إلى I/O الرقمية .



مخطط الـ Schimatic

الشكل(5-1)

المنافذ المميزة بعلامة ”P“ يمكن أن تستخدم كمداخل ومخارج خارجية (محيطية). بشكل رئيس هذا يتضمن إشارات SPI-PWM-UART.

المنافذ المميزة بعلامة ”Pi“ يمكن أن تستخدم كمدخل خارجي (محيطي) ، لكن لا يستخدم كمخرج . المنافذ المميزة بعلامة DAx و CLx تكون مستخدمة من أجل TWI(two wire interface) .

ملاحظة

بسبب الخطأ على الشاشة الحريرية (نصوص و صور مطبوعة على الشريحة) تصميم 1.5v (الكمية الأولى المنتجة من الشرائح التجارية) الحرف ”P“ محذوف من المنافذ الخارجية 33-40 و 45-48 . لكن تلك الأحد عشر منفذ يمكن أن تستخدم مداخل أو مخارج محيطية PWM .

جدول وظائف المنافذ المفصل

A/D منفذ يستطيع أن يستخدم كمدخل تشابهي .

I^2C يمكن أن يستخدم كمنفذ بيانات DAx إذا كان I^2C نموذج رقم CLx , x و نبضات ساعة إذا كان I^2C نموذج رقم x .

PPSi منفذ يمكن أن يستخدم كدخل من أجل قابلية إعادة استكشاف المحيطيات (دخل ملتقط -UART-SPI) مثال: دخل من مايكرو لليويو فالمايكرو عبد واليويو سيد هنا .

PPSo منفذ يمكن أن يستخدم كخرج من أجل قابلية إعادة استكشاف المحيطيات مثال: هنا عكس الجملة السابقة

(خرج مقارن-UART-SPI) .

5V مدخل5V تسامحي ، يمكن أن يستخدم كمدخل 5V أو مخرج 5V في نمط المصرف المفتوح .

Comp مدخل يمكن أن يستخدم كمدخل رقم مقارن .

Prog مدخل يمكن أن يستخدم ICSP : يستخدم Vpp وكلا من C1/D1 أو C2/D2 أو C3/D3 والتي تكون نبضات ساعة وبيانات على التوالي .

ملاحظات: ١- كل المنافذ يمكن أن تستخدم كمدخل / مخرج رقمي (GPIO) مصرف مفتوح . ٢- كل المنافذ يمكن أن تستخدم ك "كمقاطعات عند التغير"

IOIO pin	A/D	I^2C	PPSi	PPSo	5V	Comp	Prog	PIC pin	PIC function
1								39	OSCI/CLKI/CN23/RC12
2								40	OSCO/CLKO/CN22/RC15
3			Y	Y	Y			42	RTCC/DMLN/RP2/CN53/RD8
4		DA0	Y	Y	Y			43	DPLN/SDA1/RP4/GD8/CN54/RD9
5		CL0	Y	Y	Y			44	SCL1/RP3/GD6/CN55/RD10
6			Y	Y	Y			45	RP12/GD7/CN56/RD11
7			Y	Y	Y			46	DMH/RP11/INT0/CN49/RD0
8						3D		47	SOSCI/C3IND/CN1/RC13
9			Y			3C		48	SOSCO/SCLKI/T1CK/C3INC/RPI37/CN0/RC14
10			Y	Y	Y			49	VCPCON/RP24/GD9/VBUSCHG/CN50/RD1
11			Y	Y	Y			50	DPH/RP23/CN51/RD2
12			Y	Y	Y			51	RP22/GEN/CN52/RD3
13			Y	Y	Y			52	RP25/GCLK/CN13/RD4
14			Y	Y	Y			53	RP20/GPWR/CN14/RD5
15						3B		54	C3INB/CN15/RD6
16						3A		55	C3INA/SESSEND/CN16/RD7
17								58	GD10/VBUSST/VCMPST1/VBUSVLD/CN68/RF0
18					Y			59	GD11/VCMPST2/SESSVLD/CN69/RF1
19					Y			60	GD0/CN58/RE0
20					Y			61	GD1/CN59/RE1
21					Y			62	GD2/CN60/RE2
22					Y			63	GD3/CN61/RE3
23					Y			64	HSYNC/CN62/RE4
24					Y			1	VSYNC/CN63/RE5
25		CL2			Y			2	GD12/SCL3/CN64/RE6
26		DA2			Y			3	GD13/SDA3/CN65/RE7
27						1D		4	C1IND/RP21/CN8/RG6
28			Y	Y		1C		5	C1INC/RP26/CN9/RG7
29			Y	Y		2D		6	C2IND/RP19/GD14/CN10/RG8
30			Y	Y		2C		8	C2INC/RP27/GD15/CN11

									/RG9
31	Y		Y	Y		1A	C3	11	PGEC3/AN5/C1INA/VBUSON/RP18/CN7/RB5
32	Y		Y	Y		1B	D3	12	PGED3/AN4/C1INB/USBOEN/RP28/CN6/RB4
33	Y					2A		13	AN3/C2INA/VPIO/CN5/RB3
34	Y		Y	Y		2B		14	AN2/C2INB/VMIO/RP13/CN4/RB2
35	Y (ref +)		Y	Y			C1	15	PGEC1/AN1/VREF-/RP1/CN3/RB1
36	Y (ref -)		Y	Y			D1	16	PGED1/AN0/VREF+/RP0/CN2/RB0
37	Y		Y	Y			C2	17	PGEC2/AN6/RP6/CN24/RB6
38	Y		Y	Y			D2	18	PGED2/AN7/RP7/RCV/CN25/RB7
39	Y		Y	Y				21	AN8/RP8/CN26/RB8
40	Y		Y	Y				22	AN9/RP9/CN27/RB9
41	Y							23	TMS/CVREF/AN10/CN28/RB10
42	Y							24	TDO/AN11/CN29/RB11
43	Y							27	TCK/AN12/CTEDG2/CN30/RB12
44	Y							28	TDI/AN13/CTEDG1/CN31/RB13
45	Y		Y	Y				29	AN14/CTPLS/RP14/CN32/RB14
46	Y		Y	Y				30	AN15/RP29/REFO/CN12/RB15
47		DA1	Y	Y	Y			31	SDA2/RP10/GD4/CN17/RF4
48		CL1	Y	Y	Y			32	SCL2/RP17/GD5/CN18/RF5
stat LED			Y	Y	Y			33	RP16/USBID/CN71/RF3
mclr							Vpp	7	MCLR

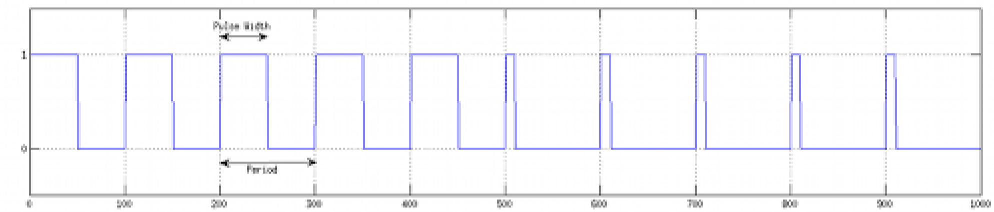
١.٥ المدخل النبضي IOIO PulseInput:

أحد أشهر الإشارات الرقمية تلك التي تعتمد فيها المعلومات على الفترات الزمنية بين التغيرات في مستويات الإشارة . ربما تكون الحساسات المغناطيسية أو البصرية هي الأسهل لتخيل ذلك عند تحسس شعاع الانعطاف ، عندما ينعطف الشعاع بشكل محدد تتغير مخارج الحساس من LOW إلى HIGH ، و عند زاوية أخرى ستتغير إلى HIGH . يتكون الشكل الموجي الناتج بانعطاف الشكل الموجي الناتج من نبضات .

كلما كان الانعطاف سريعا كانت النبضات أصغر و أقرب إلى بعضها البعض . بقياس الزمن الدوري للنبضات نجد قيمة الزمن الحقيقية التي تطلبها الشعاع لإتمام انعطاف واحد ، و بمعنى آخر نكون بذلك قد حصلنا على تردد النبضات و أصبح بإمكاننا معرفة سرعة الانعطاف.

إشارات PWM هي حالة شهيرة أخرى حيث تستخدم هذه الإشارات عرض النبضة لتمثيل قيمة خلال مدى محدد ، على سبيل المثال إن إشارات PWM في محركات DC عبارة عن سلسلة من النبضات زمنها الدوري له المدى (1-2)ms . إن نبضة 1ms عادة ما تعني (ذراع السيرفو يجب أن تتجه إلى نهاية واحدة) ، إن نبضة 2ms تعني (التوجه إلى النهاية الأخرى) ، و نبضة 1.5ms تعني المركز . إذا حاولنا فك تشفير هذه الإشارة ، فعند استقبال إشارات دخل نظام ما الخاصة بتوليد إشارات سيرفو سنهتم بقياس عرض النبضة أو المدة بين بداية النبضة و نهايتها.

الإشارة الممثلة بالشكل(6) هي إشارة PWM نموذجية حيث أن لها زمن دوري و عرض نبضة متغير ، و من الأسهل هنا الحديث عن التردد الذي هو مقلوب الزمن ، و المعنى الفيزيائي له هو عدد النبضات خلال ثانية واحدة



الشكل(6)

يتم قياس التردد و عرض النبضة في شريحة IOIO عن طريق وحدة إدخال النبضات ، هذه الوحدة لها متغيرات عديدة تؤثر في سير العملية ، لتسهيل ذلك يجب فهم آلية سير العملية الداخلية لوحدة إدخال النبضات ، لكل وحدة مؤقت خاص يعد بشكل ثابت الزمن عند نبضة ساعة معينة . عند الجبهة الصاعدة والهابطة يتم تخزين القيمة الحالية ، و يتم تحديد الزمن المنقضي بينهما عن طريق طرح الإشارتين المخزنتين الناتجتين عن كلا الجبهتين . إن للمؤقت حدا أعلى لقياس النبضة فعند الوصول لهذا الحد يبدأ العد من الصفر مرة أخرى . من المستحيل معرفة هذا الحد بدقة ولذلك عندما نحاول قياس الفترات الطويلة جدا نجد أننا حصلنا على نتائج خاطئة ، ستتم مناقشة اختيار معدل نبضات الساعة ودقة التوقيت و نمذجة التردد .

الدقة

إن الدقة هي الخيار الأول المؤثر في سير العملية حيث أن شريحة IOIO تمتلك ثلاث وحدات إدخال نبضات من نوع دقة منفردة و ثلاث وحدات أخرى من نوع دقة مزدوجة . تستخدم وحدة الدقة المنفردة مؤقت بحجم 16Bit وبذلك تستطيع قياس

النبضــات بطــول 65535 نبضــة ســاعة أمــا وحــدة الدقــة المزدوجــة فتســتخدم مؤقــت بحجــم 32Bit و تستطيع قياس النبضات حتى 4.3 مليار نبضة ساعة .

الهزاز (نبضات الساعة)

يمكــن اختيــار وحــدتي نبضــات الســاعة لتعطــي - 256KHz - 2MHz - 16MHz 62.5KHz كلمــا قــل معــدل نبضــا الســاعة زاد الــزمن المتطلــب لحــدوث الطفحــان و كلمــا زاد معــدل نبضـــات الســـاعة زادت القـــدرة علـــى قيـــاس النبضـــات الأطـــول . علـــى ســـبيل المثـــال تســـتخدم وحـــدة الدقـــة المنفـــردة معـــدل نبضـــات ســـاعة قـــدره 256KHz و تســـتطيع قيـــاس نبضـــــات بـــــأطوال تصـــــل حتـــــى (256000-65535) و تســـــاوي 0.255sec . و بالجهــة المقابلــة كلمــا كانــت نبضــات الســاعة أبطــأ كانــت دقــة القيــاس أقــل فعنــد اســتخدام نبضـــات ســـاعة بتـــردد 62.5KHz ســـوف ســـكون القيـــاس بدقـــة 1/62500=16us لـــذا سيكون الزمن 40us مقاسا في الواقع بدقة 48us الذي من الممكن أن لا يكون مقبولا

الصيغة

يــدعى البــارامتر الــذي يحــدد حــدث تعيــين بدايــة و نهايــة الــزمن المقــاس بالصــيغة ، و يمكــن أن تكــون هــذه الصــيغة إمــا نبضــة نمطهــا موجبــة HIGH أو نبضــة ســالبة LOW أو تــردد (جبهــة صــاعدة إلــى جبهــة صــاعدة أخــرى) ، و عنــد قيــاس التــردد مــن الممكــن قيــاس زمــن نبضـــة واحـــدة ، زمـــن 4 نبضـــات متعاقبـــة أو زمـــن 16 نبضـــة متعاقبـــة ، و الخيـــاران الأخيــران يســميان بنمذجــة التــردد حيــث يســمحان لنــا بنقــل مــدى التــردد المقــاس و زيــادة دقيــة القيــاس . و كنتيجــة لعــدة قياســات يســاعد بتنعــيم الضــجيج . نلاحــظ أنــه بزيــادة عامــل النمذجــة يصــبح حــد المؤقــت أكثــر دقــة عنــد القيــاس بتخفــيض التــردد المقــاس . و مــن الملاحــظ أنــه باســتخدام API عنــد نمذجــة التــردد تبقــى الطــرق كمــا هــي لإعــادة الــزمن لنبضــة واحــدة و هــي (قســمة الــزمن المقــاس علــى عــدد عوامــل النمذجــة) . ولتلخــيص الصــيغ المحتملــة يعطينــا الجــدول التــالي المواصــفات لكــل خيــار محتمــل لهــذه البــارامترات . هــذا الجــدول خــاص بعمليــة الدقــة المنفــردة و للحصــول علــى الجــدول الخــاص بالدقــة المزدوجــة (نضــرب زمــن أطــول نبضــة بــ 65535 و بشــكل معــاكس نقســم علــى نفــس القيمــة للحصــول علــى التــردد) . بشــكل ملائــم تــم تصــنيف الجــدول الســابق علــى أســاس أطــول نبضــة و للحصــول علــى الدقــة الأمثليــة يجــب اختيــار أخفــض صــف يكــافئ أعلــى زمن نبضة متوقع

Clock	Scaling	Resolution	Longest pulse	Lowest frequency
62.5KHz	1	16us	1.048s	0.95Hz
250KHz	1	4us	262.1ms	3.81Hz
62.5KHz	4	4us	262.1ms	3.81Hz
250KHz	4	1us	65.54ms	15.26Hz
62.5KHz	16	1us	65.54ms	15.26Hz
2MHz	1	500ns	32.77ms	30.52Hz
250KHz	16	250us	16.38ms	61.0Hz
2MHz	4	125ns	8.192ms	122.1Hz

16MHz	1	62.5ns	4.096ms	244.1Hz
2MHz	16	31.25ns	2.048ms	488.3Hz
16MHz	4	15.6ns	1.024ms	976.6Hz
16MHz	16	3.9ns	256us	3.906KHz

نمط 16Mhz مناسب للعديد من التطبيقات .

الاستخدام:

استخدام تعليمات المنفذ النبضي لشريحة يويو ضمن الجافا كالتالي:

```
IOIO.openPulseInput(n)
```
لفتح المنفذ النبضي رقم n في الشريحة

```
PulseInput pulse = ioio.openPulseInput(pinNum, mode);
```
لفتح المنفذ النبضي pinNum وتحديد نمطه من بين الأنماط الثلاثة التالية :

PulseMode.POSITIVE ليعد جبهة صاعدة

PulseMode.NEGATIVE ليعد جبهة هابطة

PulseMode.FREQ ليعطي تردد كافة النبضات

```
float pulseSeconds = pulse.getDuration();
```
ضع قيمة النبضات الزمنية الفترة الزمنية (قياس عرض النبضات) الداخلة للمنفذ في متحول pulseSeconds

```
float freqHz = pulse.getFrequency();
```
ضع قيمة التردد المقاسة بمتحول freqHz

```
float pulseSeconds = pulse.waitPulseGetDuration();
```
ضع قياس عرض النبضات بدون خسارة النبضة البدائية بمتحول pulseSeconds

```
pulse.close();
```
إنهاء استخدام المنفذ كمنفذ نبضي وإمكانية استخدامه لأغراض أخرى .

١.٦ المدخل التشابهي IOIO AnalogInput:

نحتاج إلى قياس الجهد في كثير من الحالات وغالبا ليس الجهد بحد ذاته هو ما نهتم به، ولكن بعض الكميات الفيزيائية التي تم تحويلها إلى جهد بواسطة حساسات . إن أبسط الأمثلة هي المقاومة المتغيرة عند وصلها بطريقة ما ، فهي تمثل إحدى مخارج المقاومة المتغيرة التي تتغير تبعاً لزاوية تغير المقاومة على أحد أطرافها ولكن هذا مثال بسيط من الأمثلة المتنوعة على الحساسات لقياس القيم الفيزيائية . تمتلك IOIO 16 منفذ معروفة بالمداخل التشابهية قادرة قياس مستويات الجهد من (0 - 3.3) فولت بدقة تصل الى 3mV .

يجب أن نتأكد بأن لا تصل قيم الجهد خارج هذا المجال على هذه المنافذ ، وإلا سيحدث ضرر في البنية الصلبة . في بعض الحالات عندما يكون الجهد المقاس خارج هذا المجال يمكن أن نضع جسر مقاومات لقياس هذا الجهد . وفي حالات أخرى نحتاج إلى دارة مضخم .

داخلياً تقوم IOIO بأخذ عينات زمنية من الجهد المطبق وإرسال هذه العينات إلى جهاز أندرويد . عند قراءة هذه العينات يكون الخيار إما بأخذ آخر قيمة تم الحصول عليها أو بأخذهم عينة تلو الأخرى ،تحاط المنافذ التشابهية بمربعات سوداء خلف الشريحة.

الاستخدام

يتم استخدام المنافذ التشابهية كمداخل عبر واجهة AnalogInput . إن هذه العملية تستجيب للقيم الفيزيائية على المنفذ في الشريحة ، مضبوطة للعمل في النمط التشابهي . إن واجهة AnalogInput تحدد باستعداء

```
AnalogInput  in = ioio_.open AnalogInput(pinNum) ;
```

Pinnum تحدد رقم المنفذ في المدخل التشابهي حيث يتم تحديد قيمة المدخل التشابهي باستدعاء التابع AnalogInput

```
Float value = in.read();
```

بإرجاع القيمة بين الصفر والواحد ، يتم تغطية القيمة الدخلة من الجهد المسموح . إذا تم استدعاء هذا التابع بشكل فوري يقوم بحجب بضع أجزاء من الثانية بعد فتح المنفذ إذا كانت القيم المقاسة بالفولت مناسبة أكثر ، نستخدم :

```
Float value = in.getvoltage ();
```

عند الانتهاء من استخدام هذا المنفذ نستدعي التابع

```
In.close();
```

لإعادة المنفذ الى الحالة (floating) و إمكانية إعادة فتح هذا المنفذ في أي وضع ، يصبح هذا المنفذ غير قابل للاستخدام بعد استخدام هذا التابع

تخزين قيم القراءات

دائماً ما كنا نهتم في الاستخدامات السابقة بآخر قيمة للإشارة ، في بعض الأحيان من المهم أن نحدد العينات التي تم أخذها في كل الفترات الزمنية (مثال رسم المخططات وقياس مدى تغير الإشارة) . في هذه الحالة لا نستطيع أن نسمح للعينات بأن تضيع إذا كنا نأخذ القراءات ببطء شديد ولا نستطيع بالسماح بأخذ العينة مرتين إذا كنا نأخذ القراءات بسرعة . غالباً نأخذ حلاً وسطياً بتشكيل قائمة، لهذا الهدف نقدم تخزين قيم القراءات .

و لاستخدام تخزين قيم القراءات نستدعي التابع

```
In.SetBuffer(capacity);
```

تحدد capacity عدد العينات الغير مقروءة حيث يقوم المكدس بتخزينها قبل حدوث حالة الطفحان. فيستخدم المكدس الأصغر ذاكرة أقل ، ولكن يجبرنا على أخذ القراءات بتواتر أسرع لتجنب الطفحان . وعند حدوث حالة الطفحان يتم حذف العينات الأقدم

لقراءة العينات من المكدس نستدعي التابع :

```
Float Sample = in.readBuffered();
```

أو

```
Float Volts = in.getVoltageBuffered();
```

سوف تحجب هذه الطرق حتى توفر عينة واحدة على الأقل في المكدس وبعد ذلك يتم إعادة هذه العينة . إذا كان الحجب غير مرغوب به يمكن استدعاء التابع التالي أولاً

```
Int samples = in.available ();
```

حيث تقوم هذه العينات الموجودة في المكدس عند حدوث الطفحان بتحديد عدد العينات التي تم فقدانها باستدعاء التابع In.getOverFlowCount() ولمعرفة معدل أخذ العينات نستدعي التابع

```
in.getsampleret()
```

ملاحظة : إن منافذ المداخل التشابهية تأخذ العينات بمعدل 1000 عينة للثانية الواحدة (1khz) في بعض التطبيقات (مثل قياسات الإشارات الصوتية) وهذا المعدل غير مقبول .

في شريحة IOIO لا يوجد حد فيزيائي لأخذ العينات (في الحقيقة إن الحد الفيزيائي هو 500khz لكل القنوات المفتوحة) ويمكن تطويره في المستقبل .

١.٧ المداخل والمخارج الرقمية في IOIO:

المداخل والمخارج المنطقية (على مستوى المنطق) هي اللبنات الأساسية في بناء الانظمة الرقمية الصلبة و يشار إليها باسم GPIO . إشارة المستوى المنطقي لها حالتين LOW-HIGH بالترتيب 0 أو 1 ، true او false .

تختلف التمثيلات الفعلية لهذه الحالات من حيث الجهد والتيار ويعتمد بشكل كبير على الدارات التي تولدلهم ، بالمقابل تأثر على الدارات التي تستهلكهم ، ويجب فهم تلك التمثيلات الممكنة المختلفة (أو على الأقل تلك الشائعة منها) .

كل منافذ IOIO يمكن أن تكون بمثابة مداخل رقمية أو مخارج رقمية ، وهذا يتوقف على اختيار المشغل في مجال البرمجة ، كما هو موضح أدناه . حالة الــ LED توضح وضع المنفذ ، ويمكن التحكم باعتباره خرج رقمي .

الخرج الرقمي

عندما تستخدم على نحو الإدخال ، في أوضاع التشغيل الافتراضية الخاصة ، منافذ IOIO تمثل مستويات منطقية على النحو التالي :

- الحالة المنطقية LOW سوف توثر على المنفذ بفرض الجهد 0V "الأرضي" على كل ما هو مرتبط به .
- الحالة المنطقية HIGH سوف توثر على المنفذ بفرض الجهد 3.3V على كل ما هو مرتبطة به .

العديد من الالكترونيات الرقمية ستقبل بهذا التمثيل، والقراءة الصحيحة للمستويات LOW كـ LOW والمستويات HIGH كـ HIGH. وكمثال على ذلك، الـ LED متصل بين منفذ الخرج الرقمي (مصعد LED)، من خلال المقاومة 220Ω إلى الأرض وسوف يضيء عندما تكون حالة المنفذ HIGH .

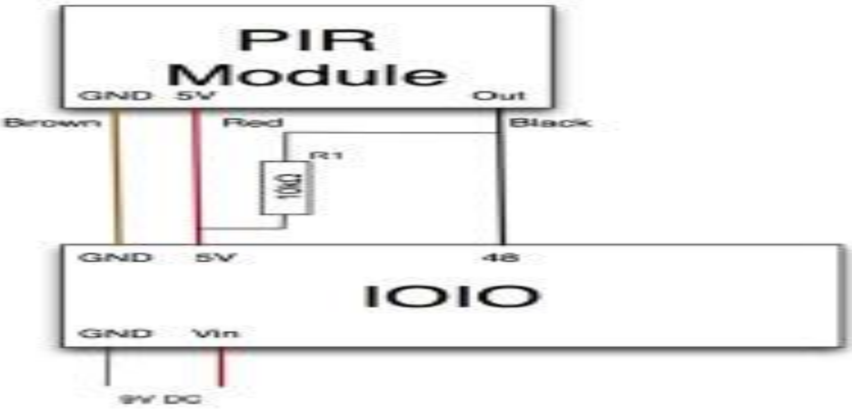
بالنسبة لبعض العناصر عملية "المصرف المفتوح" تكون مطلوبة ،في نمط العمل "المصرف المفتوح" وضع المنافذ سيمثل المستويات المنطقية بشكل مختلف قليلاً :

- **الحالة المنطقية LOW** سوف توثر على المنفذ بفرض الجهد 0V "الأرضي" على كل ما هو مرتبط به ، مثلما هو الحال في الوضع العادي .
- **الحالة المنطقية HIGH** سوف توثر على المنفذ إلى عدم فرض أي شيء على كل ما هو مرتبطة به ، كما لو كانت منفصلة فيزيائياً .

وضع المصرف المفتوح يمكن أن يكون مفيدة في العديد من الحالات. والأكثر شيوعا هو أن بعض المكونات الإلكترونية تتطلب 5V / 0V إشارة بدلا من 3.3V / 0V.

الطريق إلى تحقيق ذلك هو في غاية في البساطة. أولا لهذا الغرض تحتاج إلى تحديد منفذ بسماحية 5V . إذا حاولت تنفيذ هذا الأسلوب على منفذ غير متسامح 5V هذا سيشكل خطر بشكل دائم على المنفذ المستخدم أو الشريحة بأكملها. وتتم تسمية المنافذ 5V المتسامح على الجزء السفلي من الشريحة مع الدارات حول المنفذ. يمكنك أيضا استشارة جدول المنافذ السابق.
الدائرة التالية: التي تتألف فقط من مقاومة واحدة "سحب المقاومة" ، سوف تؤثر في الجهد للتبديل بين 5V / 0V عندما تستخدم بنمط "المصرف المفتوح" . بشكل آخر، والوضع أقل

شيوعا لاستخدام "المصرف المفتوح"، هو عندما يربط منافذ خرج عدة مكونات إلى نقطة واحدة



، تدعى "bus "

وبما أننا لا نريد منفذين في محاولة لوضع مستويات الجهد المختلفة على شبكة واحدة، لئلا تتلف بعضها البعض، نحن نستخدم كل منهما في وضع المصرف المفتوح ، واستخدام مقاومة سحب واحدة ما يصل إما إلى 3.3V أو 5V . أما عندما يكون هناك منفذ او اكثر في مجموعة خطوط BUS " نظام سحب "، فإن جهد مجموعة الخطوط يكون جهدها 0V ، كل المنافذ ستراقب خطوط "BUS" وستكشف اي تغير بها .إذ كافة المنافذ بالمستوى المنطقي HIGH ، أي عملياً تفصل ، والجهد BUS يحصل به "انسحاب "من قبل المقاومة على كل الجهد المتصل بها.

وكشرح بسيط عن التيار فعندما تفرض IOIO جهد معين على واحد من منافذها، ما هو الحمل المتصل بهذا المنفذ و الذي يحدد كمية التيار التي يجب ان تقدمها شريحة IOIO . الأحمال المختلفة لديها علاقات مختلفة بين الجهد و تحصل على التيار الذي تحتاجه بالإجبارمالم يكن هناك عائق لها.

على سبيل المثال، المقاومة لديها علاقة خطية بين الجهد والتيار واعتمادا على مقاومته للمنفذ..

هناك حدود على كمية التيار في منافذ IOIO سواء منبع أو مصرف .الحد الأقصى الحالي الموصى بها هو حوالي 20MA. عند محاولة زيادة التيار اكثر من 20MAسيكون هناك خطر على المنفذ او على الشريحة بأكملها .للنصيحة أي المقاومة أصغر من 150ohm على 3.3V ستتجاوز هذه المشكلة ويفضل ٢٢٠ أوم .

في الحياة الحقيقية ، وعلى المرء أن يفهم نوع الحمل المتصل لتحديد قيمة التيار الذي سوف يستهلك .

فكقاعدة عامة الربط مع المكونات المنطقية (ميكروكنترولر، وأجهزة الاستشعار، وضوابط هيدروليك، الخ) هي عموماً ليست مشكلة الأنواع الحمل الأخرى، مثل المحركات والمصابيح عالية الطاقة تستهلك الكثير من التيار (وعادةً ما تتطلب جهد عالي جدا) . مثل هذه الأحمال سوف تحتاج الى دارة قيادة ، مع المواصفات المناسبة .

الدخل الرقمي

عندما تستخدم المداخل كمداخل ، منافذ IOIO لن تفرض أي جهد على عنصر يرتبط بها ، وإنما تقوم باستشعار مستوى الجهد وتقدم تقرير للحالة وفقا لذلك.

- عندما يتم الكشف عن 0V، المنفذ يقرأ LOW
- عندما يتم الكشف عن 3.3V، المنفذ يقرأ HIGH . المنافذ ذات السماحية 5v، سوف تسمح بمرور التغذية 5V، وأيضاً التعامل معها على مستوى HIGH .
- عندما لا يفرض اي جهد على المنفذ، المنافذ يمكن أن يقال عنها:

  - لن تؤثر على منفذ بأي شكل من الأشكال .فإن قيمة القراءة تكون عشوائية نوعا ما ، ولا يمكن الاعتماد عليها .وهذا ما يسمى وضع "عائم"، وهذا هو الوضع الافتراضي للمنافذ الإدخال .
  - يرفع المنفذ إلى القيمة 3.3v. ، بحيث ان القيمة ستقرأ HIGH حتى يقوم المستخدم بفرض القيمة 0V على المنفذ مثلا ٦ .وهذا ما يسمى نمط " الرفع"
  - يسحب المنفذ إلى القيمة 0V بحيث ان قيمة ستقرأ LOW حتى يقوم المستخدم بفرض القيمة 3.3V (أو 5V في منافذ المحدد)على تلك المنافذ .وهذا ما يسمى نمط "السحب" .

عمليات الرفع والسحب مفيدة لمجموعة كاملة من التطبيقات . وعدة استخدامات متكررة لتوصيل المفتاح فاستخدام المنفذ في وضع الرفع .عندما يفتح المفتاح ، المنفذ يقطع اتصاله ، وسوف يعمل في وضع الرفع مما يوصل المنفذ إلى شعور HIGH .وعندما يتم إغلاق المفتاح، سيضع على المنفذ القيمة 0V، وأنها سوف تقرأ LOW . وبالمثل يمكن للدخل العمل في وضع سحب وربط المفتاح بين المنفذ و3.3V .

استعمال الخرج الرقمي

يتم ذلك باستخدام منافذ IOIO كمخارج رقمية من خلال واجهة DigitalOutput. في هذه الواجهة تتوافق حالتها مع الحالة الفيزيائية للمنفذ على الشريحة ، تكوين العمل في وضع الخرج الرقمي .ويتم الحصول على الحالات DigitalOutput بواسطة استدعاء واحد من مدخلات التحميل الزائد للتابع .IOIO.openDigitalOutput() ، ابسط شكل هو :

```
DigitalOutput out = ioio_.openDigitalOutput(pinNum);
```

هذا يقوم بفتح رقم المنفذ pinNum للخرج الرقمي ، في الحالة الطبيعية (غير حالة المصرف المفتوح) ، يكون المنفذ بحاجة إلى الوقت من أجل إجراء عملية الطلب لفتح المنفذ من أجل الخرج الرقمي في نمط عمل المصرف المفتوح يجب أن تكون التوابع بالشكل التالي :

```
DigitalOutput out =
ioio_.openDigitalOutput(pinNum,DigitalOutput.Spec.Mode.OPEN_DRAIN,
startValue);
```

startValue تحدد القيمة البدائية (LOW أو HIGH) عندما يتم تحديدها كخرج . يمكن أن تستخدم لتجنب الحالات اللحظية الغير مرغوبة للمنفذ . بالعودة إلى Javadocs لواجهة شريحة IOIO من أجل عدة متغيرات متوفرة في هذه الطريقة .

الليد الضوئي الموجود على الشريحة "ليد الحالة" يستخدم في منفذ الخرج الرقمي العادي . ويفتح باستخدام :

```
DigitalOutput led = ioio_.openDigitalOutput(IOIO.LED_PIN);
```

الليد الضوئي يكون قد تمت تغذيته لذلك يغير حاله من LOW إلى وضع التشغيل HIGH ومن ثم يطفئ .

بإلغاء DigitalOutput، المنفذ يمكن أن يوضع على الحالة LOW من خلال استدعاء :

```
out.write(false);
```

أو إلى الوضعية HIGH باستدعاء :

```
out.write(true);
```

عندما تنتهي من هذه التعليمة ، لن تستخدم المنفذ ، فاستدع :

```
out.close();
```

لإعادة المنفذ للحالة " floating " و يكون قابل لإعادة فتحه في نفس أنماط العمل المختلفة ، المنفذ يكون بدون جدوى بعد الاستدعاء

استعمال الدخل الرقمي

يتم ذلك باستخدام منافذ IOIO كما في حالة المداخل الرقمية عبر واجهة DigitalInput . في هذه الواجهة تتوافق حالتها مع الحالة الفيزيائية للمنفذ على الشريحة ، تكوين العمل في وضع الدخل الرقمي . ويتم الحصول على الحالات DigitalInput بواسطة استدعاء واحد من مدخلات التحميل الزائد للتابع .IOIO.openDigitalInput() و أبسط شكل هو:

```
DigitalInput in = ioio_.openDigitalInput(pinNum);
```

حيث يقوم بفتح رقم المنفذ pinNum للدخل الرقمي ، في الحالة الطبيعية (غير حالة المصرف المفتوح) ، يكون المنفذ بحاجة الى الوقت من أجل إجراء عملية الطلب ، هذا المنفذ لن يكون من أجل أي شي آخر ، في الترتيب ، لفتح المنفذ من أجل الدخل الرقمي في نمط عمل المصرف المفتوح يجب أن تكون التوابع بالشكل التالي :

```
DigitalInput in =  ioio_.openDigitalInput(pinNum,DigitalInput.Spec.Mode.PULL_UP);
```

PULL_DOWN بالعودة إلى واجهة Javadocs توفر شريحة IOIO من أجل عدة متغيرات في هذه الطريقة .عوضاً عن DigitalInput التي ألغت قيمة المنفذ يمكن أن تكون جاهزة بواسطة :

boolean value = in.read();

يقوم المنفذ بإرجاع false للمستوى المنطقي المنخفض و true للمستوى المنطقي العالي ، هذه الطريقة قد تمنع المنفذ من الإخراج لبضع ميلي ثانية إذا استدعى على الفور بعد أن يكون المنفذ مفتوح .

هناك استخدام آخر مثير للاهتمام هو DigitalInput وهو الحجب حتى يتم الحصول على قيمة معينة تكون ملغية ، بخلاف توزع الأقطاب دورياً ، وربما في النبضات الضيقة المفقودة ، من المفضل أن نستخدم :

in.waitForValue(true);

من أجل الانتظار لتصبح القيمة HIGH .وبالمثل مع false للمستوى المنطقي المنخفض ،

وهذا مفيد جدا، على سبيل المثال، للانتظار حتى يتم الضغط على زر من قبل المستخدم.
عندما يتم الانتهاء من استخدام المنفذ، نستدعي :

in.close();

من أجل إرجاع المنفذ إلى حالة " floating "وربما نكون قادرين على إعادة فتحه في نفس النمط أو في نمط مختلف .يصبح المنفذ عديمة الفائدة بعد هذا الاستدعاء . و لايمكننا القيام بأي شيء مع المنفذ لإعادة القراءة ضمن البرنامج .

استخدام تعليمات منافذ PWM في اليويو بورد :

```
PwmOutput pwm = ioio.openPwmOutput(pinNum, freq);
```

رقم المنفذ pinNum المخرج لـ PWM وfreq طول تردد النبضة المخرجة بالـHz .

و هذه التعليمة يتم استخدامها بالنمط الطبيعي ليس ٥فولط مصرف مفتوح .

أما تعليمة نمط مصرف مفتوح ٥ فولط هي التالي

```
PwmOutput pwm = ioio.openPwmOutput(new DigitalOutput.Spec(pinNum,
Mode.OPEN_DRAIN), freq);
```

لتحديد الـdc الـDutyCycle بالتعليمة

```
pwm.setDutyCycle(dc);
```

إذ أنَّ **dc** تأخذ قيم بين (٠-١) المقابلة لـ (٠%-١٠٠%) طبعاً يمكن كتابة رقم ضمن المجال السابق على أنه **dc** مثل ٠.٥ وتعني ٥٠% .

```
pwm.setPulseWidth(pw);
```

pw تمثل وضع وفرض عرض نبضة مباشرة بالمايكرو ثانية فهنا ١٠٠٠ تعني ٢ ميلي ثانية

```
pwm.close();
```

للإغلاق

أحب أن أنوه لكم في جافا الأندرويد يويو ال ١٠ كيلو هرتز تقابل قيمة ١٠ مايكرو ثانية .

١.٨ الحساسات السعوية في IOIO:

إن الحساسات السعوية تحظى بانتشار واسع هذه الأيام بداية بشاشات اللمس وانتهاء بأزرار المصاعد ومن أشهر تقنيات تحديد اللمسة البشرية هي الحساسات السعوية حيث أن هذه التقنية مبنية على أن الجسم البشري له سعة و التي يمكن قياسها وتحديدها عند التلامس (حتى عند الاقتراب) مع سطح التلامس الحساس فمن أهم ميزات استخدام الحساسات السعوية هي تحديد اللمس بدون الحاجة للضغط . ومن مساوئها عدم القدرة على العمل عند ارتداء القفازات أو أي عنصر غير موصل .

إن مبدأ الحساسات السعوية بسيط جداً . عند الدفع بشحنة كهربائية Q في مكثف يتغير الجهد V تبعاً لهذه الشحنة . إن النسبة Q/V (كمية الشحنة الواجب تطبيقها لتغير الجهد) تدعى بالسعة وتعطى بالقيمة C . تقاس السعة بوحدات تدعى فاراد ،حيث أن الفاراد الواحد هو شحنة واحدة لكل فولت ، مثلاً لقياس سعة دارة من الممكن قيادة كمية شحنة معروفة خلالها (عن طريق قيادة تيار ثابت معروف خلال وحدة زمن معروفة) ، و بعد ذلك نراقب التغير في الجهد . يتصل أي سطح موصل بمنفذ حساس سعة ليستطيع بدوره أن يقوم مقام حساس السعة .

من أبسط الطرق لصناعة الأزرار بأشكال معينة هي ورق الألمنيوم ففي بعض الحالات يحتاج الشكل إلى إخذ التأريض بالحسبان كما سيتم وصفه لاحقاً .

تحتوي IOIO على 16 منفذ من الممكن أن يكون حساس سعوي و هي نفسها المنافذ المستخدمة كمداخل تشابهية وارقامها 46 - 31 . وتصل إلى 16 زراً أو حتى 64 زراً عن طريق منظومة Charlieplexing 8×8 .

أخذ العينات

يتم قياس المنافذ الـ 16 مرة كل 16ms عن طريق قياس الجهد وفي بقية الوقت يتم وصل المنفذ بالأرضي لتفريغ شحنته

الترشيح

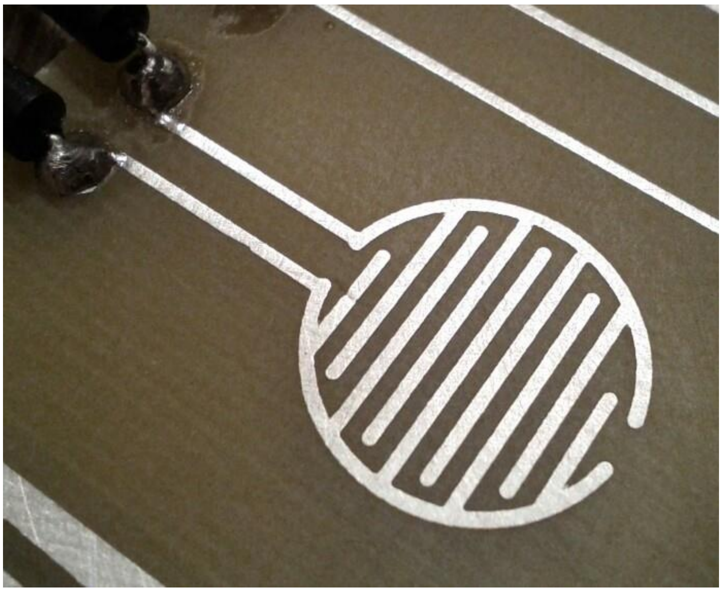
إن إشارة الجهد مشوشة جداً بشكل طبيعي و هي بحاجة لترشيح للحصول على نتائج جيدة ، إن شريحة IOIO تحتوي على مرشح مدمج . من الممكن ضبطه لكي يكون سريع ومشوش أو بطيء وناعم (أو أي نقطة بين ذلك) . يتم التحكم بهذا المرشح بضبط قيمة الإشارة ذات البعد زمني . سوف نحصل من هذا الزمن على معدل تغير نموذجي للإشارة المرشحة على سبيل

المثال سوف نحصل على استجابة أفضل بضبطه على قيمة 20-50ms ، مناسبة للتفاعل البشري .

التأريض

إن مشاركة الخط الأرضي بين شريحة IOIO و حساس اللمس يزيد من فعالية السعة المقاسة بشكل ملحوظ . لن يكون هناك أي مشكلة عند وصل شريحة IOIO بالخط الأرضي في المأخذ الجداري أو عند وصلها بكمبيوتر مكتبي لكن عند التشغيل بنمط البطاريات سنحصل على نتائج في القياسات أصغر بكثير من ان يكون ملاحظتها . كحل لهذه المشكلة يمكن تصميم حساس مثل الحساس المستخدم باللمس وهذا الحساس بحد ذاته من الممكن ان يلاحظ في تأريض IOIO ، مثل التصميم التالي بالشكل(7)

بالنظر إلى هذه الصورة نلاحظ وجود موصلين منفصلين في هذا الزر يتم وصل أحدهما إلى الخط الأرضي والآخر إلى منفذ الحساس السعوي ، إن حجم هذا الحساس أصغر من لمسة الإصبع فعند لمسه يتم لمس كلا الموصلين



الشكل(7)

المدى والدقة

إن سعة صغيرة سوف تسبب جهدا قدره 3.3 فولت خلال عملية الشحن بتحديد السعة الدنيا بشكل فعال يمكن أن يتحسس الحساس وإن أقل قيمة لهذه العتبة هي 27pf . إن شريحة IOIO نفسها لها سعة داخلية ، هذه السعة أعلى بشكل طفيف من سعة الحمل وهي صغيرة بشكل كاف حتى تهمل في الحياة العملية .

من ناحية أخرى سوف تسبب السعات الكبيرة تغيراً في الجهد كنتيجة لشحن ثابت . بما أن دقة أخذ العينات للجهد محددة بالتالي ستحدد القيم الفعالة للسعات المحتملة العلية التي يتم قياسها ولكن قبل أن نصل إلى هذا الحد سوف نخسر الدقة فكضبط سابق نحصل على دقة 10% من اجل سعة قدرها 2700 pf (عند تجاهل الضجيج) ، وهذه ليست مشكلة عند كشف اللمسة البشرية .

الاستخدام

يتم استخدام منافذ شريحة IOIO كحساسات سعوية عبر تعليمة Capcence. إن هذه التعليمة تستجيب لمنفذ فيزيائي واحد تم ضبطه ليعمل في هذه الوضعية . يتم استدعاء هذا التابع عن طريق التعليمة () IOIO.OpenCapSence و أبسط شكل لها هو

```
CapSence capSence=IOIO_.openCapSence(pinNum);
```

حيث تستخدم هذه التعليمة المرشح الافتراضي بقيمة 25ms وليجتاز قيم المرشح الافتراضية نستخدم :

```
CapSence capSence=IOIO_.openCapSence(pinNum,FilterTime);
```

حيث أن FilterTimeمقاسة بـ ms

وعند تحديد CapSence يمكن الحصول على قيمتها عن طريق استدعاء

```
Float pacoFarade = CapSence.read();
```

بإعادة القيمة بواحدة pf تقوم هذه الطريقة بحجب بعضاً من العينات إذا تم استدعائها بشكل فوري بعد أن يتم فتح المنفذ .غالباً ما يراد تحديد السعات التي تتجاوز جهد العتبة التي تشير للمسة بالطريقة التي نقوم بها . ولتجتنب ذلك نقوم باستخدام الطرق التالية

```
void waitForClick() throws ConnectionLostException,
InterruptedException {
  capSense.waitUnder(50);  // wait until there is no touch detected.
  capSense.waitOver(60);   // wait until touch is detected.}
```

إن تعليمة waitUnder سوف تقوم بالحجب حتى قياس سعة أقل من 50pF ، تقوم تعليمة waitOver() بالحجب حتى تصل القيمة المقاسة إلى 60pF

إنَّ العتبات مختلفة بالهدف وتستخدم للتقليل من التداخل الناجم عن اللمس المتكرر.

أخيرا: عند تغيير ثابت الترشيح الزمني نقوم باستدعاء التابع

```
Capsence.setfiltercoef(timeInMs);
```

الذي يضبط الوقت في المرشح . إن القيمة 25ms مناسبة بشكل طبيعي لتحديد اللمسة بشكل سريع و معقول.

٢.١ الأردينو ARDUINO ولماذا؟

Arduino هو أشبه بكومبيوتر صغير الحجم بإمكانه التفاعل و التحكم في الوسط المحيط، تقنياً هو منصة برمجية Platform مفتوحة المصدر تتكون من متحكم إلكتروني Micro-Controller.

وسنتعرف في هذا الفصل على مكوناته وطريقة التفاعل معه.

٢.١.١ نبذة تاريخية عن إطلاق مشروع الأردوينو وتطوره:

تبدأ القصة في عام ٢٠٠٥ في مدينة ايفريا Ivrea الإيطالية عندما اجتمع فريق من مهندسي الإلكترونيات ليناقشوا سبل تعليم طلاب الجامعة والمعاهد التقنية علوم المتحكمات الدقيقة أو ما يعرف باسم ال Microcontrollers.

كان الهدف من اللقاء إيجاد أسلوب سهل وغير مكلف لإدخال هذا النوع من العلوم في الدراسة الجامعية ويكون متاح للهواة في ذات الوقت دون أن تكون هناك قيود أو تراخيص مكلفة يحتاجون لشرائها لاستخدام البرمجيات التي تستخدم في برمجة المتحكمات الدقيقة في هذا الوقت ...

و من هنا تبدأ رحلة نشأة أردوينو Arduino.

قام كل من "ماسيمو بانزي Massimo Banzi " بالتعاون مع "دايفيد كوارتيليس David cuartielles " وجاينلو كامارتينو Gianluca Martino " بإطلاق مشروع أردوينو ايفريا Arduino of Ivrea وتمت تسمية المشروع باسم أشهر شخصية تاريخية في المدينة وكان الهدف الأساسي للمشروع هو عمل بيئة تطوير للمتحكمات الدقيقة بصورة مفتوحة المصدر 100 % وتضمن هذا المشروع عمل بيئة تطوير برمجية للمتحكمات الدقيقة Integrated Development Environment وتكون مجانية في ذات الوقت كما تضمن عمل لوحات تطويرية Development Boards صغيرة الحجم بتكلفة بسيطة يمكن للطلاب والهواة والتقنيين تحمل سعرها.

بدأ تطوير المشروع اعتماداً على مشروع سابق اسمه Wiring platform والذي قد نجح في تحقيق بعض أهداف مشروع أردوينو الأساسية مثل المصدر المفتوح ولكن لم يصل إلى المستوى الذي تمناه فريق التطوير من الناحية البرمجية والتصميم العتادي Hardware Design لذلك تم البدء في العمل على تطوير اللغة البرمجية المستخدمة في لغة Processing وسميت هذه

اللغة الجديدة باسم Arduino C وهي مشابهة جداً للّغة البرمجية ++C من ناحية طريقة كتابة الأوامر وتنسيق الأكواد.

تم إطلاق أول لوحة تطويرية لأردوينو في أواخر عام ٢٠٠٥ واعتمدت على شريحة ATmega168 من العائلة المشهورة AVR والتي تنتجها شركة Atmel للشرائح الإلكترونية ، وسميت هذه اللوحة باسم Arduino Serial V.1 .

وما أن تم نشر المشروع حتى توالت الكثير من التطويرات على التصميمات الهندسية للّوحات الإلكترونية وبيئة التطوير من مهندسين وهواة مختلفين حول العالم وتم إصدار العديد من اللّوحات الأحدث مثل :

Arduino UNO, ArduinoMEGA, Arduino Llypad , Arduino Duemilanove , Arduino FIO

وكان القاسم المشترك في كل الإصدارات السابقة هي أنها تعتمد على شرائح المتحكمات الدقيقة من عائلة AVR التي تنتجها شركة Atmel وحقق المشروع نجاحاً باهراً حتى وصل عدد اللوحات الإلكترونية المباعة إلى ٣٠٠.٠٠٠ قطعة في مختلف دول العالم.

في عام ٢٠٠٩ تعرض أردوينو لنقلة نوعية جديدة جاءت على يد فريق LeafLabs والذي قام بتطوير لوحة أردوينو جديدة تسمى The Maple Board تعمل بشرائح الكترونية من عائلة ARM cortex والتي نقلت أردوينو إلى أفق جديد من السرعة والقوة حيث تتميّز جميع شرائح ARM باحتوائها على معالجات عالية السرعة تمتلك قوة معالجة تبدأ من ٧٢ ميجا هرتز وتصل في بعض الإصدارات إلى ١.٥ جيجا هرتز .

وبالرّغم من كل التطورات السريعة في لوحات الأردوينو المختلفة إلا أن التطوير لم يتوقف عن حدود ARM Cortex بل وصل إلى شرائح " مصفوفات البوابات المنطقية القابلة للبرمجة Field-Programmable Gate Arrays (FPGA)" في عام ٢٠١٠ تم نشر مشروع مفتوح المصدر اسمه Papilio Arduino وكان الهدف منه هو صناعة لوحة أردوينو تعمل بشرائح من نوع FPGA ويمكن برمجتها بلغة Arduino C.

تتميز شرائح FPGA بالسرعة الفائقة في أداء عمليات المعالجة بصورة متوازية وفك تشفير البيانات Decoding لذلك تستخدم في الكثير من التطبيقات الصناعية التي يكون فيها عامل الوقت أمراً حاسماً في تصميم المنتج النهائي لكن يعيب هذه الشرائح عدة أمور تجعلها مقتصرة على المحترفين والشركات فقط.

الأول : تكلفة لوحات التطوير الخاصة بها والتي يبدأ سعرها غالباً من ١٠٠ دولار وأكثر.

ثانيا: صعوبة برمجة الشرائح حيث تستخدم لغات برمجية معقدة في تشغيل هذه الشرائح مثل VHDL ولغة Verilog.

جاء مشروع papilio Arduino ليحل هذه المشكلات ويقدم لوحة تطوير جديدة إلى عائلة لوحات أردوينو وتم إصدار papilio Arduino IDE وهي نسخة معدلة من برنامج Arduino IDE لجعل لغة أردوينو تتناسب مع شرائح FPGA كما تم إصدار لوحات papilio boards للبيع من منتصف عام ٢٠١٠ بأسعار تبدأ من ٤٠ دولار فقط.

بعد مشروع papilio Arduino أصبحت عائلة أردوينو تحتوي على لوحات تطويرات تعمل تقريباً بكافة التقنيات التي تم ابتكارها مثل AVR وشرائح ARM Cortex وحتى شرائح FPGA.

وفي النهاية يمكننا القول أن مشروع أردوينو صنع أكبر مجتمع تقني من المهندسين والفنيين والهواة يعملون على تطوير الأفكار والمشاريع المتعلقة بالتحكم الآلي بصورة تشاركية وتفاعلية حول العالم اعتماداً على استخدام لوحات إلكترونية بتقنيات مختلفة ومع ذلك يتم برمجتها جميعاً بلغة برمجية واحدة ومجانية ومتاحة للجميع .

٢.١.٢ تعريف لوحة أردوينو:

هو بورد إلكتروني مفتوح المصدر Open Hardware لتطوير الكثير من الأفكار والمشاريع المتعلقة بالتحكم الآلي بصورة سهلة وبسيطة عن طريق استخدام لغة برمجة مفتوحة المصدر Arduino C ويتم برمجة المتحكم الموجود على البورد باستخدام برنامج خاص يسمى: Arduino IDE: Integrated Development Environment ويمكن تحميله مجاناً من موقع الأردوينو الرسمي:.http://arduino.cc يمثل الشكل (١.٢) لوحة أردوينو.

٢.١.٣ ما معنى أن أردوينو مفتوح المصدر ؟

يعني أنه يمكن الإطلاع والتعديل على التصميمات الهندسية والشيفرات المصدرية source codes لكل من بوردات أردوينو المختلفة arduino boards و arduino IDE بما يناسب المستخدم ، ويمكن أيضا تطوير لغة برمجة Arduino C بحرية تامة والاطلاع على الشيفرات المصدرية الخاصة بها كما أن كل هذه الميّزات والبرمجيات مجانية تماماً على عكس بعض البيئات التطويرية مثل Micro C والذي تتطلب شراء رخصة مكلفة تصل أحيانا إلى آلاف الدولارات لاستخدامها .

ما يميز بوردات الأردوينو عن باقي البوردات التطويرية للمتحكمات الدقيقة الأخرى

Micro controllers Development Boards هو مدى السهولة في التعامل معها وبساطة اللغة البرمجية والتي عُمل على تطويرها منذ عام ٢٠٠٥ حتى الآن، ولقد تم اشتقاق لغة أردوينو البرمجية من لغة processing ولغة "سي C" والتي تُعد أساس لغات البرمجة الحديثة.

وقد تم تطوير أردوينو ليناسب جميع المستخدمين بدءاً من الهواة وانتهاءً بالمشاريع المتطورة والدليل أنه هناك مميزات جبارة تجعله على قائمة المتحكمات الدقيقة وهي إمكانية دمجه في مشاريع يتم برمجتها بلغات هندسية متطورة مثل MATLAB , labview ولغة Java حيث يوجد مكتبات برمجية جاهزة للغة ال MATLAB وال Java خاصة بالتعامل مع أردوينو ، كما يمكننا أيضا استخدام لغة VB.NET في التواصل مع بوردات أردوينو المختلفة .



خطأ! لا يوجد نص من النمط المعين في المستند.

الشكل (*2 - 1*) لوحة أردوينو

٢.٢ أنواع بوردات أردوينو:

تختلف بوردات أردوينو عن بعضها البعض من ناحية عدد المخارج والمداخل والتي تحدد عدد الأجهزة التي يمكن التحكم بها وعدد الحساسات Sensors التي يمكن دمجها مع البورد وكذلك نوع المتحكم الدقيق وسرعة المعالج الموجود بداخلها وإمكانية تبديلها أم لا، ومن هذه الأنواع :

Arduino UNO , Arduino Mega , Arduino Nano , Arduino mini , Arduino Lilypad , Arduino Demulive , Boarduino .

وفي مايلي بعض أنواع دارات أردوينو ومايميزها عن بعضها :

٢.٢.١ Arduino Mega :

دارة إلكترونية تستخدم متحكم ATMEL (Atmega2560).وتوفر هذه الدارة منافذ لتوصيل المكونات الإلكترونية إلى المتحكم مباشرة عن طريق ٥٤ (مدخل/مخرج) من النوع الرقمي من هذه ال ٥٤ يوجد ١٦ مدخل تشابهي و ٤ منافذ تسلسلية ، أيضا تحتوي الدارة على هزاز كريستالي بتردد ١٦ ميغاهرتز بالإضافة الى مدخل USB من أجل التوصيل مع الحاسب وهناك مأخذ للطاقة منفصل ، و زر إعادة الإقلاع . ويمثل الشكل (٢.٢) دارة Arduino Mega .



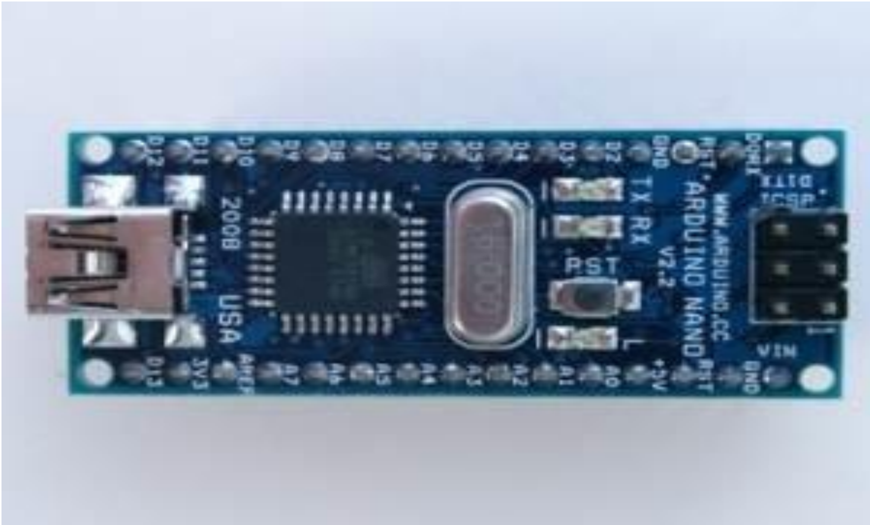
الشكل (٢.٢) دارة *Arduino Mega*

٢.٢.٢ Arduino Nano:

دارة إلكترونية صغيرة الحجم و كاملة تستخدم في برمجة متحكم

ATmega328 (Arduino Nano 3.x)

ATmega168 (Arduino Nano 2.x) ، إنها تفتقر لمأخذ الطاقة وتعمل فقط مع كابل -Mini USB B. ويمثل الشكل (٣.٢) دارة Arduino Nano



الشكل (٣.٢) دارة *Arduino Nano*

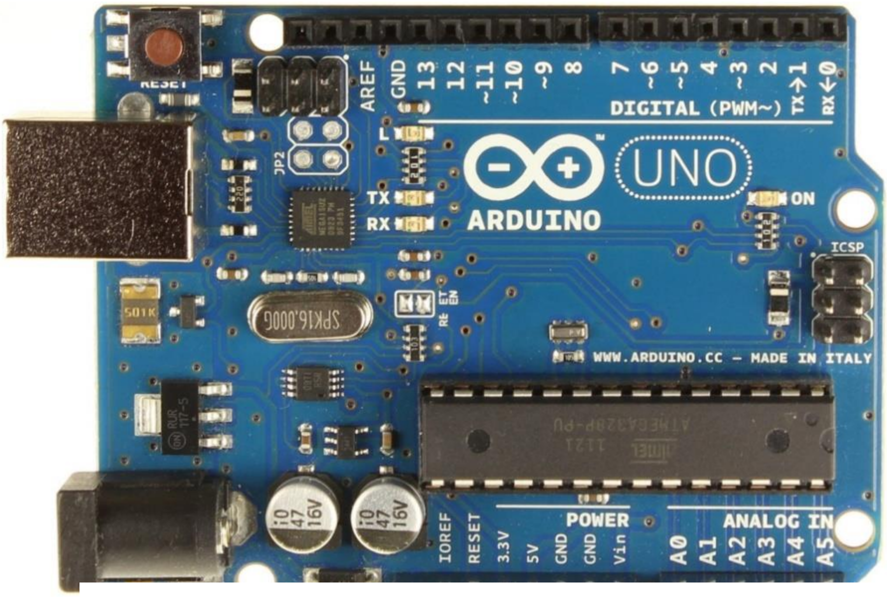
٢.٢.٣ Arduino Uno:

سنستخدم في تجربتنا بورد Arduino UNO، وهي دارة إلكترونية صغيرة تستخدم في برمجة متحكمة من شركة اتمل ATmega328P-PU وتوفر هذه الدارة منافذ لتوصيل المكونات الإلكترونية الى المتحكم مباشرة عن طريق ١٤ (مدخل / مخرج) من النوع الرقمي Digital In/Out و ٦ مداخل من النوع التشابهي.

هذه ال ١٤ يوجد ٦ يمكن استخدامها كمخارج PWM أو ما يعرف بالتعديل الرقمي المعتمد على عرض النبضة (Pulse-Width modulation) تعديل نبضي مطالي .

أيضا تحتوي الدارة على هزاز كريستالي Crystal Oscillator بتردد ١٦ MHz، بالإضافة الى مدخل USB من أجل التواصل مع الحاسب و هناك مدخل للطاقة منفصل، بالإضافة الى ICSP header والذي يعتبر طريقة إضافية لبرمجة المتحكمة و هي لاتزال موصولة بالبورد (بخلاف ال USB) ويمكنك أن تعتبر بورد أردوينو هذه بورد تطوير وبرمجة مصغرة ومهيئة للاستخدام المباشر Development Board فهي تقريباً تحوي كل ما تحتاج لكي تعمل عليها سواء عن طريق منفذ ال USB أو عن طريق مصدر خارجي للطاقة مثل البطارية.

والشكل (٤.٢) يمثل دارة أردوينو UNO



الشكل (2 - ٤) دارة اردوينو

٢.٣ الأردوينو وإمداد الدارة بالطاقة Power UP :

يتم إمداد الدارة بالطاقة إما من خلال منفذ ال USB فقط أو عن طريق استخدام مصدر خارجي للكهرباء كمحول AC/DC ليمد الدارة بالجهد اللازم للعمل أو حتى عن طريق بطارية ٩ فولت أو ٦ بطاريات تفرعية / ١.٥ فولت /حيث يتم توصيل طرفي البطارية إلى مدخل الأرضي Gnd وال Vin في الدارة .

تستطيع الدارة العمل على جهد يتراوح بين ال [٦-٢٠]فولت ، لكن يجب الانتباه حيث إذا قمنا بتأمين جهد أقل من ٧ فولت فإن مخرج المتحكم Pin 5V قد لا يستطيع تأمين جهد خرج يبلغ ال ٥ فولت المطلوب وقد يؤدي إلى عدم استقرار الدارة ، أما اذا قمنا بتزويد الدارة بجهد أعلى من ١٢ فولت فإنه قد يؤثر على عنصر تنظيم الجهد voltage regulator ويؤدي الى ارتفاع درجة حرارته مما يؤدي الى تلف البورد ، لذا فإن مجال الجهد الذي يفضل استخدامه هو من ٧ الى ١٢ فولت

٢.٣.١ مخارج ومداخل الطاقة الكهربائية للمتحكم أردينو:

VIN : جهد الدخل عندما نستخدم مصدر خارجي ، يمكننا تأمين الجهد من خلال هذا المدخل.

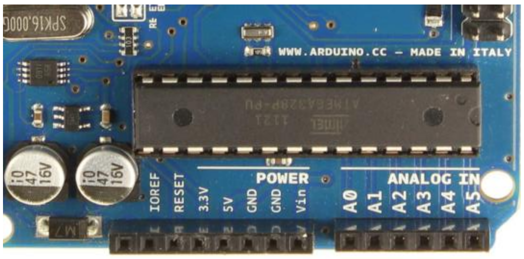
V5 : جهد منتظم يستخدم لتأمين الطاقة للعناصر المستخدمة على الدارة وسوف نستخدمه لتوفير الطاقة للقطع الإلكترونية التي سنضيفها ، قد يأتي هذا الجهد من خلال Vin عبر منظم جهد

داخلي أو تأمينه من خلال منفذ ال USBأو أي مصدر جهد منتظم بقيمة ٥ فولت ولا يعطي أكثر من ٥٠٠مل أمبير .

V3.3 : مصدر للجهد بقيمة ٣.٣ فولت مؤمن من قبل منظم الجهد الداخلي للدارة وأقصى قيمة لسحب التيار من خلال هذا الخط هو ٥٠ ميلي أمبير.

GND : الخط الأرضي.

ويمثل الشكل (٥.٢) مخارج ومداخل الطاقة الكهربائية للمتحكم



الشكل (٥.٢) مخارج ومداخل الطاقة الكهربائية للمتحكم

٢.٤ شريحة المتحكم ATmega 328P-PU:

المتحكمات الدقيقة أشبه بوحدة حاسب صغير الحجم ويحتوي المتحكم الدقيق ATmega328P-PU على معالج بسرعة ١٦ ميجا هرتز وذاكرة كلية تساوي ٣٢ kilo byte بالإضافة إلى :

Boot loader: السوفت وير المسؤول عن كيفية فهم الدارة للغة Arduino C.

SRAM : تعتبر الذاكرة المستخدمة في تسجيل المتغيرات بصورة مؤقتة.

Flash Disk: مساحة تخزينية تستخدم في تخزين البرنامج الذي سنكتبه لتشغيل المتحكم.

EEPROM : الذاكرة المسؤولة عن تسجيل بعض المتغيرات بصورة دائمة داخل المتحكم وتظل محتفظة بقيمتها حتى بعد فصل الكهرباء.

٢.٤.١ أهم مميزات شريحة المتحكم ATmega 328p-pu:

معالج عرض ال DataBus له ٨Bit بين الذاكرة والمعالج.

البنية المعمارية له تستخدم فلسفة Risc ، أي تعليمات أقل وبنية أكثر تعقيداً.

٣٢ مسجل عرض bit ٨ تستخدم لأغراض عامة.

يعمل على تردد ١٦MHZ وله ثلاث ذواكر :

EEPROM=1K Byte

FLASH=32K Bytes

SRAM=2K Bytes

يحتوي عل مؤقتين وعدادين مع مقسم للتردد ويعمل على compare mode:

أحدهما 16bit مؤقت وعداد والآخر 8bit.

6 قنوات PWM مبدل عرض النبضة.

٨ قنوات مبدل تشابهي رقمي ADC .

يبرمج بشكل تسلسلي.

مصادر مقاطعات داخلية وخارجية.

٢٣ قطب دخل وخرج.

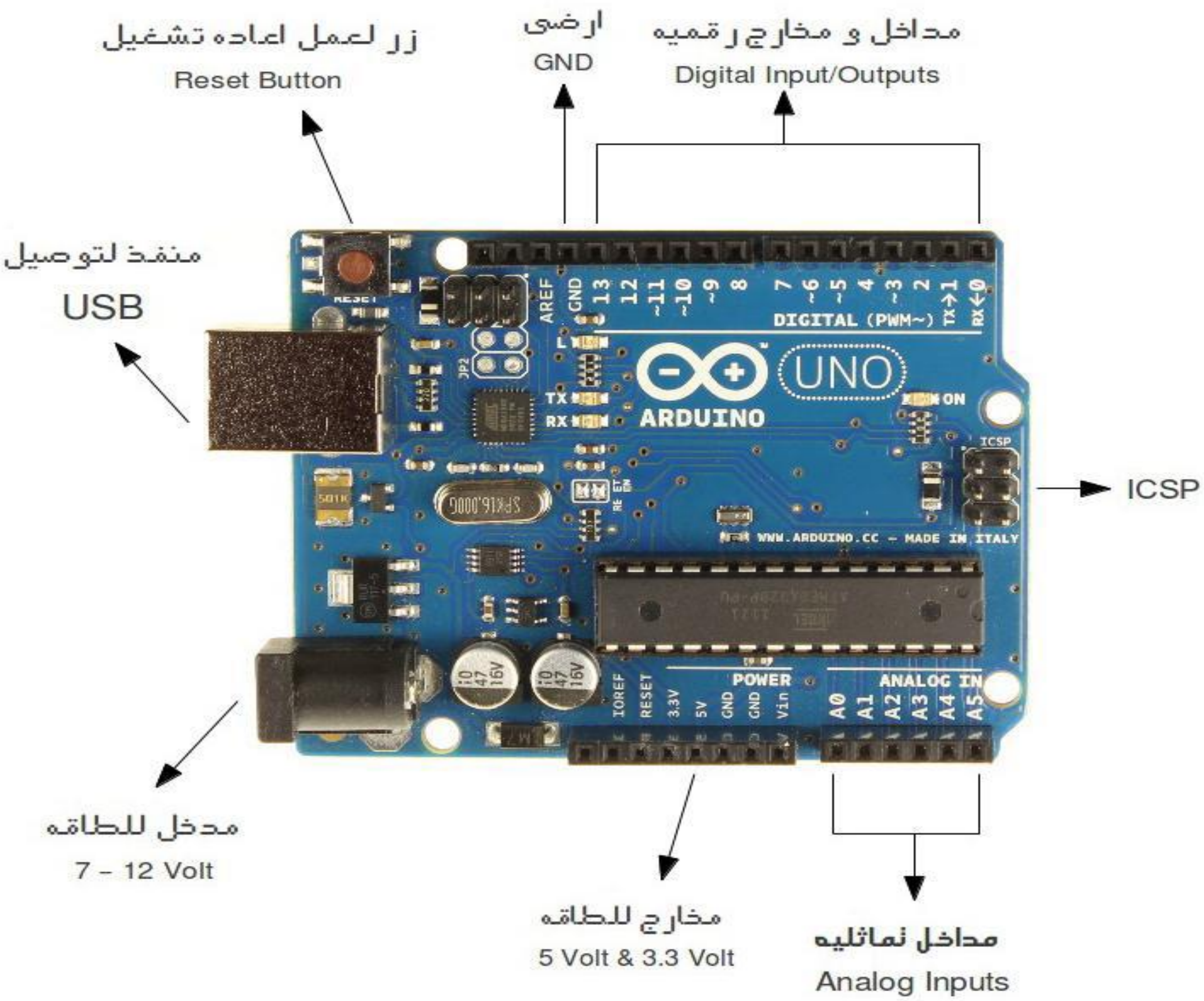
يعمل المتحكم على جهد يتراوح من V1.8 حتى V٥.٥ .

٢.٤.٢ مداخل ومخارج التحكم (I/O) input & output Pins :

يمكن تخصيص الخطوط الرقمية الأربعة عشر (Digital Pins 14) كمداخل أو مخارج وذلك باستخدام الأوامر البرمجية ، وتعتمد هذه الخطوط على جهد أقصاه ٥ فولت وكل خط يمكن أن يؤمن سحب للتيار بحدود ال ٤٠ ميلي أمبير.

ويوجد ٦ خطوط دخل/خرج يمكن الاستفادة منها بخاصية PWM وهي ٣، ٥، ٦، ٩، ١٠، ١١.

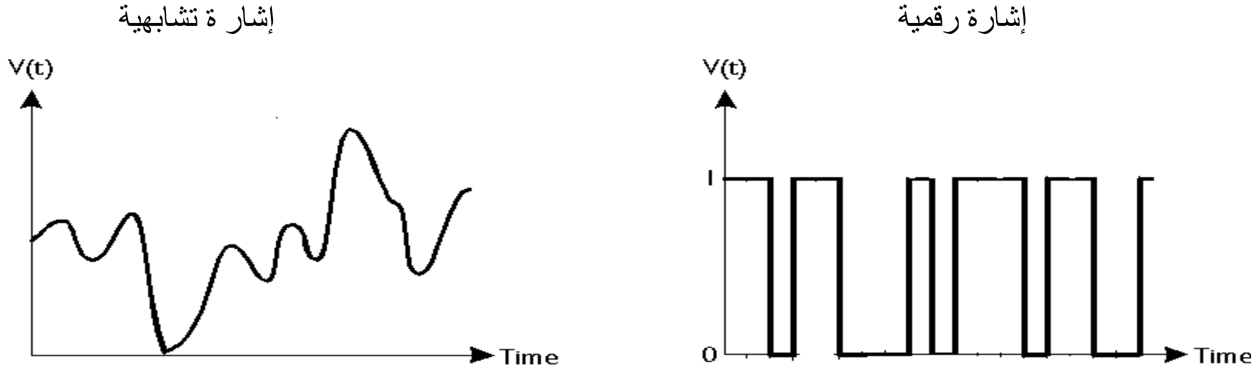
وهناك ٦ خطوط دخل تشابهية Analog ومعنونه من A0إلى A5، بشكل افتراضي تستطيع هذه المداخل قياس جهد من صفر حتى ٥ فولت. ويمثل الشكل (٦.٢) مداخل ومخارج التحكم input & output Pins (I/O)



الشكل(٦.٢) مداخل ومخارج التحكم *input & output Pins (I/O)*

٢.٤.٣ الدخل والخرج التشابهي:

يعرف الدخل أو الخرج التشابهي بأنه أي فرق جهد تبدأ قيمته من صفر وقابل للتغير دائماً وليس له حدود قصوى معينة، على عكس الدخل الرقمي والذي إما أن يكون ٥ فولت High (1) وإما أن يكون صفر فولت Low (0). ويمثل الشكل (٧.٢) الإشارة الرقمية والإشارة التشابهية.



الشكل(٧.٢) الإشارة الرقمية والإشارة التشابهية

٢.٤.٤ ماهي أهمية الإشارات التشابهية Analog signals:

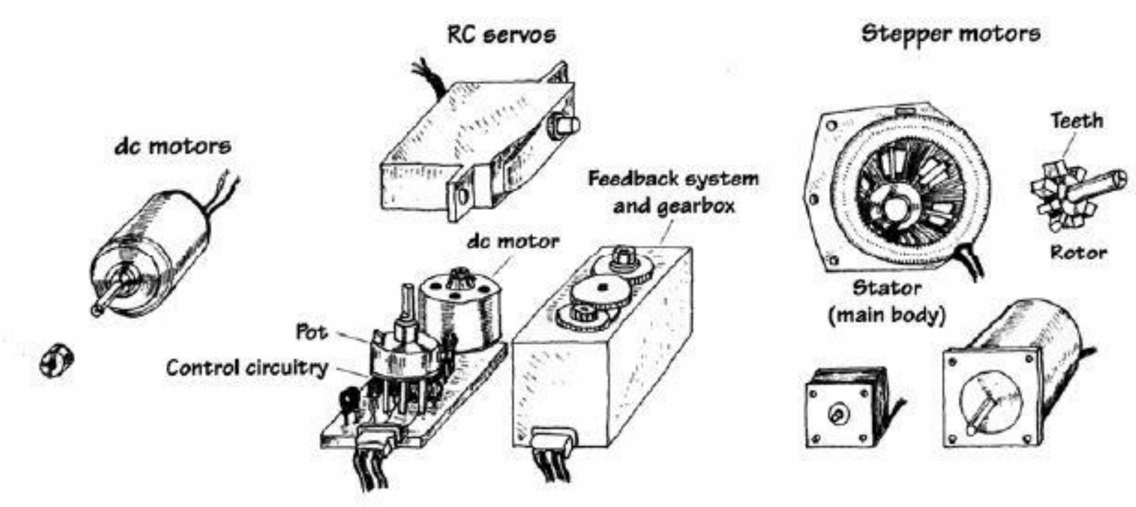
يعبر الدخل التشابهي عن الإشارات المتغيرة مثل فرق الجهد الناتج من حساس sensor معين نتيجة التغير في عامل معين، ويمكننا أن نستغل هذه الظاهرة في قياس العوامل البيئية باستخدام حساسات مناسبة وتحويل القياسات إلى إشارة كهربائية تشابهية قابلة للقياس حيث هنالك العديد من الحساسات التي تستخدم بكثرة مثل : الحرارة ، الضوء ، الرطوبة، سرعة الرياح، القوة ، العزم ، التسارع ، المجال المغناطيسي... الخ

٢.٤.٥ المداخل التشابهية في الأردوينو:

تحتوي شريحة ATmega 328P-PU المستخدمة في بورد Arduino Uno على ٦ مداخل تستطيع قياس مقدار فرق الجهد الكهربائي المتغيير (دخل تشابهي) وهي مرقمة من A0 حتى A5 ويمكنها قياس فرق جهد بدقة تصل إلى ٤.٨ ميللي فولت وهذا يعني تقريباً خمسة أجزاء من الألف جزء من واحد فولت وهي دقة عالية تكفي للكثير من التطبيقات الدقيقة .

٣.١ المحركات وطرق التحكم بها دارات القيادة :

ربما تكون أحد أهم الأشياء التي تدعو للتفكير هي استخدام الإلكترونيات في تحريك الأجهزة الميكانيكية، وتوجد ثلاث أدوات أساسية تحرك الأشياء ميكانيكيًا وهي محركات التيار المستمر، و الـ RC SERVO



الشكل (12)

الشكل (12) نماذج لمحركات تيار مستمر ، RC Servo

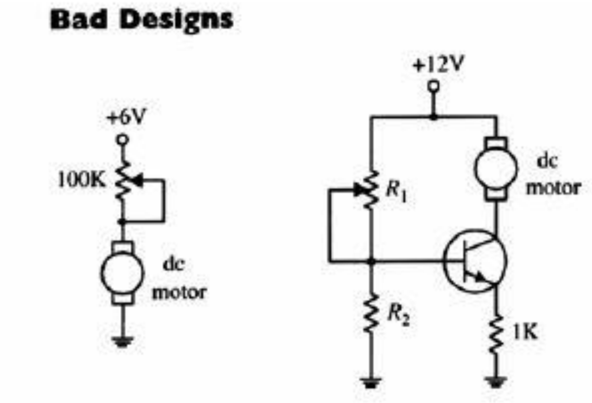
١.٣ ـ محركات التيار المستمرDC :

محركات التيار المستمر هي أجهزة يتم التحكم بها كهربائيًا ولها سلكان لتوصيل التغذية، ويكون لمحرك التيار المستمر جذع (Shaft) دوار يمكن ان يركب عليه دولاب (Wheel) أو ترس تغير سرعة (Gear) أو مروحة أو غيرها .تدور محركات التيار المستمر عددًا كبيرًا من الدورات في الدقيقة (rpm) بالنسبة لحجمها، ويمكن أن تدور باتجاه عقارب الساعة أو بعكس عقارب الساعة ويكفي عكس قطبية تغذية المحرك لتدويره باتجاه معاكس .عند الدوران بسرعات منخفضة يكون عزم المحرك منخفضًا ولا يمكن التحكم بدقة بالوضع الزاوي لجذع المحرك مما يجعل استخدام هذه المحركات للتحكم بالموضع غير ممكن . تتوفر محركات التيار المستمر بحجوم وأشكال مختلفة وأغلب أنواعها تدور بسرعات تتراوح بين (3000) و (8000) دورة في الدقيقة (rpm) عند جهد تشغيل محدد يتراوح بين (1.5) و (24) فولت . وتحدد الجهة الصانعة قيمة جهد التغذية الذي يعمل عنده المحرك بكفاءة عالية .يمكن تخفيض جهد تغذية المحرك قليلا لتخفيض سرعته، ولكن إذا انخفض جهد التغذية إلى ما دون (50%) من الجهد الاسمي فإن المحرك يتوقف عن الدوران .وكذلك إذا زاد الجهد المطبق على المحرك بنسبة (30%) عن الجهد

الاسمي فإن المحرك يسخن) ترتفع حرارته (وقد يتعطل .إن أفضل طريقة للتحكم بسرعة دوران المحرك هي استخدام ما يسمى تعديل عرض النبضة (Pulse Width modulation)) وفيها ينتقل المحرك بسرعة بين حالات (on) و (off) ويتم التحكم بسرعة دوران المحرك حسب عرض النبضة وحسب الفترة الزمنية الفاصلة بين النبضات .يستهلك المحرك (غير المحرك) تياراً صغيراً ، ولكن عند تحميل المحرك فإن التيار الذي تستهلكه الملفات الداخلية للمحرك يزداد بشكل هائل (وقد تتجاوز زيادة التيار 100% بالمقارنة مع حالة عدم وجود حمل) . تعطي الشركات الصانعة ما يسمى (Stall Current) لمحركاتها، أي تيار التوقف، وهو التيار الذي يستهلكه المحرك لحظة توقفه، وإذا كان هذا التيار غيرُ معطى فبالإمكان قياسه بواسطة مقياس تيار حيث يوصل مقياس مع المحرك ويوصل حمل إلى جذع المحرك أو يطبق عليه عزم وتسجل قيمة التيار لحظة توقف المحرك . يُعطى معدل عزم المحرك المستمر (torque rating) وهو مقدار القوة التي يستطيع المحرك بذلها على الحمل، والمحرك ذو معدل العزم العالي يستطيع بذل قوة على حمل مثبت بشكل مماسي لذراعه الدوار أكبر من القوة التي يبذلها محرك بمعدل عزم أقل . يعطى معدل عزم المحرك عادة بــ (oz/in) (g/cm) (lb/ft) .

التحكم بسرعة محركات التيار المستمر

توجد طريقة بسيطة للتحكم بسرعة محرك التيار المستمر عن طريق تحديد التيار المار عبر المحرك باستخدام مقسم جهد كما في الدارة من الشكل (2-13) وحسب قانون أوم ينخفض التيار بزيادة المقاومة وتنخفض سرعة المحرك، ولكن استخدام مقسم جهد للتحكم بسرعة دوران المحرك غير فعال (inefficient) لأن زيادة المقاومة تؤدي إلى زيادة تبديد الاستطاعة عليها، وتبدد الاستطاعة على المقاومة بشكل حرارة وهذا شيء غير جيد لأنه يستهلك استطاعة مصدر التغذية ويؤدي إلى تسخين المقاومة، فمن الممكن أن تنصهر المقاومة .يمكن التحكم بسرعة دوران محرك التيار المستمر باستخدام مضخم ترانزستور كما في الدارة الشكل (2-13) ولكن هذه الطريقة قليلة الفعالية والكفاءة أيضًا لأن زيادة تيار المجمع بزيادة تيار القاعدة تؤدي إلى زيادة الاستطاعة المبددة على الترانزستور وقد يؤدي ذلك إلى انهيار الترانزستور .



الشكل (2.13): دارات بسيطة للتحكم بسرعة دوران محرك DC.

الشكل(13)

تستخدم طريقة للتحكم بسرعة محرك التيار المستمر تشبه الطريقة الواردة في مصادر التغذية التقطيعية ، وتمتاز هذه الطريقة بتوفير الطاقة وبعدم تعريض العناصر للانصهار .تعتمد هذه الطريقة على إرسال نبضة تيار إلى المحرك ويتم التحكم بسرعة الدوران بتغيير عرض هذه النبضة وكذلك بتردد نبضات التيار .وبذلك فإن العناصر الموجودة في الدارة لا تخضع لإجهاد تياري دائم . يبين الشكل (14) ثلاث دارات بسيطة تستخدم لتأمين نبضات التحكم بالسرعة :

في الدارة الأولى يولد هزاز استرخاء يعمل على ujt سلسلة من النبضات التي تقود ثايرستور Scr (off) (on)

يضبطّ تردد النبضات وبالتالي سرعة دوران المحرك بواسطة دارة RC لأن التردد يتغير بتغير الثابت الزمني ،

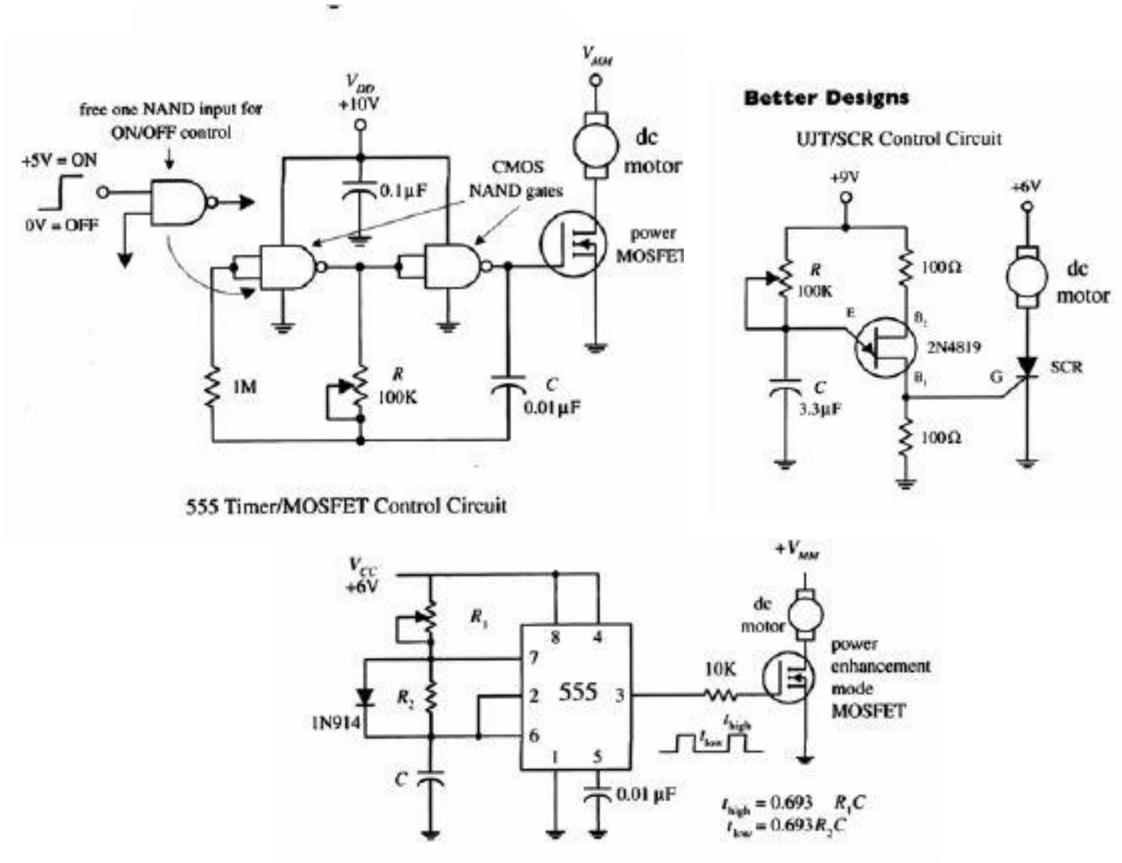
في الدارة الثانية يستخدم زوج من بوابات NAND لتكوين هزاز استرخاء ويستخدم ترانزستور MOSFET

من النوع المعزز لقيادة المحرك .وكما في الدارة السابقة تتعلق سرعة المحرك بالثابت الزمني RC للدارة .يمكن ترك أحد مداخل بوابة NAND اليسارية مفتوحًا واستخدامه كمدخل تحكم on/off وصل هذا المدخل مع دارة منطقية CMOS لتأمين اشارة القيادة .

في الدارة الأخيرة من الشكل (14) تستخدم دارة 555 كمولد نبضات مربعة ويؤمن وصل الديود كما في الشكل مع المقاومة R2 نبضات خرج بدورة مشغولية منخفضة (Low duty cycle).

يتعلق تردد نبضات clock بقيم العناصر R1 , R2 و C . وتعطى الأزمنة التي تكون خلالها النبضات في حالة on او off بالعلاقات التالية :

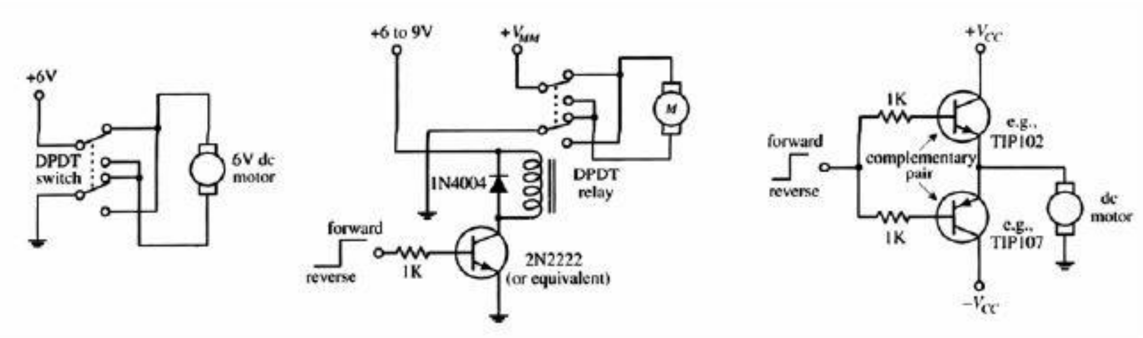
ton = 0.693R1C   toff = 0.693R2C



الشكل(14)

التحكم الاتجاهي بمحركات التيار المستمر :

يجب عكس قطبية الجهد المطبق على أطراف المحرك للتحكم بجهة دورانه وأبسط طريقة لتغيير قطبية الجهد تتم باستخدام مفتاح DPDT كما في الدارة اليسارية من الشكل (15) ويتم يدويًا اختيار الوضع المناسب للمفتاح بحيث يدور المحرك بالجهة المطلوبة .يمكن استخدام مفتاح DPDT مقاد بواسطة حاكمة في الدارة الموجودة في الوسط .إذا كنت لا تحب الحواكم استخدام دارة دفع - جذب ترانزستورية كما في الطرف اليمني من الشكل (15) تحوي هذه الدارة ترانزستورين أحدهما NPN والآخر PNPلهما نفس معدل الاستطاعة ونفس عامل تكبير التيار (Similar beta).عند تطبيق جهد high (+5V) على القواعد ينتقل ترانزستور(On) إلى حالة(npn) ويمر تيار من موجب التغذية عبر الترانزستور إلى المحرك فالأرض ويدور المحرك بجهة وعند تطبيق (0V) على القواعد ينتقل الترانزستور (PNP) الى حالة (NPN) ويمر تيار من الأرض عبر المحرك والترانزستور PNPإلى سالب المصدرVCC ويدور المحرك بجهة معاكسة للجهة الأولى.

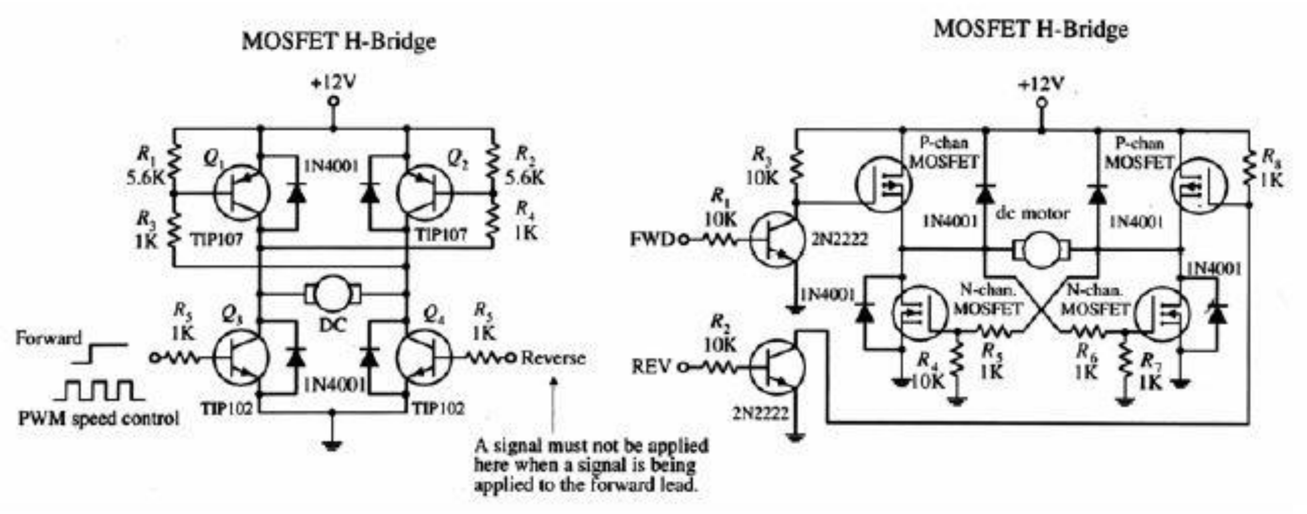


الشكل(15)

توجد دارة أخرى شائعة جدًا للتحكم بجهة دوران محرك التيار المستمر (وكذلك للتحكم بالسرعة) و هي دارة الجسر H .

و يبن الشكل (16) نموذجين بسيطين من دارة الجسر H ، الدارة اليسارية مصممة على ترانزستورات ثنائية القطبية، أما الدارة اليمينية فمصممة على ترانزستورات MOSFET لجعل المحرك يدور بالاتجاه الأمامي

(forward direction) يطبق جهد (V5+) على مدخل الدوران الأمامي، ولا تطبق أية إشارة على مدخل الدوران العكسي(لا يسمح بتطبيق جهد على المدخلين في وقت واحد) .يتم التحكم بسرعة دوران المحرك عن طريق التحكم بعرض النبضات المطبقة على الدخل .عند تطبيق جهد على قاعدة الترانزستور (Q3) ينتقل Q3 الى حالة تمرير إلى حالة تمرير ويسمح ذلك للترانزستور Q2 بأن ينتقل أيضًا إلى حالة تمرير ويمر تيار عبر المحرك من الطرف اليميني إلى اليساري (لنفترض أن المحرك يدور في هذه الحالة بالاتجاه الأمامي) .لتدوير المحرك بعكس الاتجاه تُفصل الإشارة عن دخل (Q3) وتطبق على قاعدة (Q4) فينتقل (Q4) و (Q1) معه إلى التمرير ويمر تيار عبر المحرك من اليسار إلى اليمين ويدور المحرك باتجاه عكسي يعمل جسر H المبني على ترانزستورات MOSFET بنفس الطريقة .تساعد الديودات الموجودة في الدارة على تخميد القفزات العابرة التي تنتج عن ملفات المحرك وبذلك تتم حماية باقي عناصر الدارة من التأثر بهذه الومضات .كافة الترانزستورات (ما عدا الترانزستورات ثنائية القطبية الموجودة في الدارة اليمينية) يجب أن تكون ترانزستورات استطاعية.

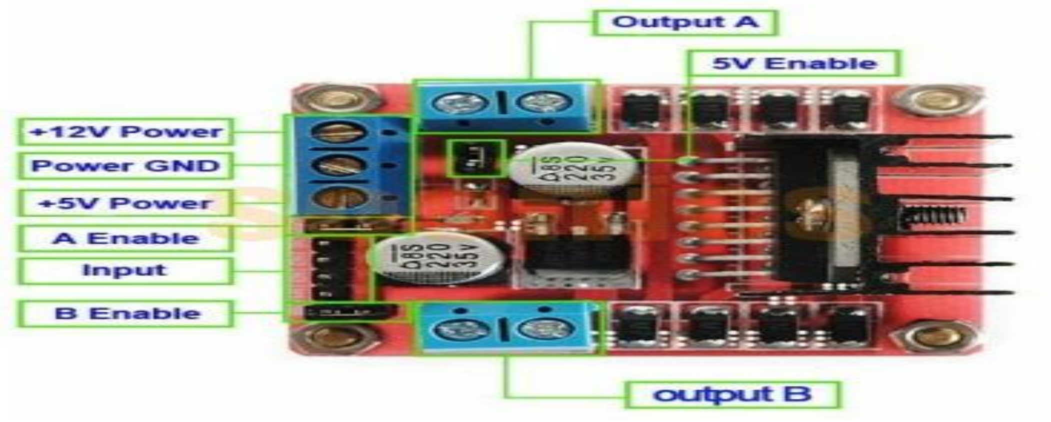


الشكل(16)

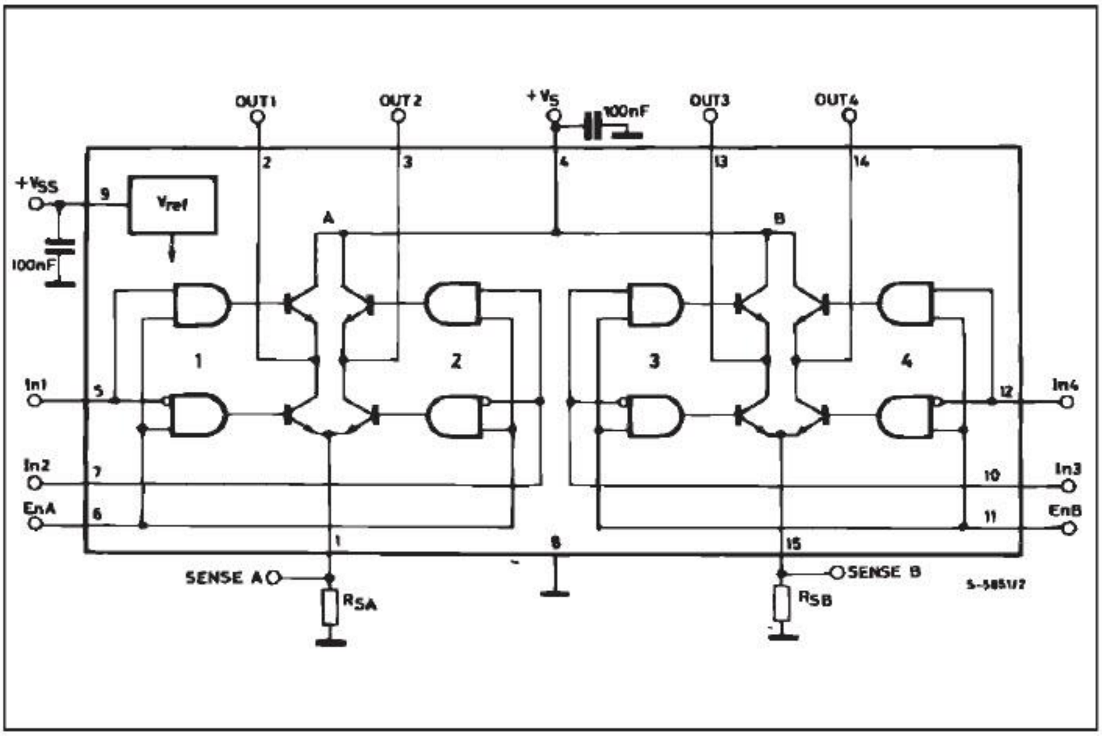
يمكن تشكيل دارات الجسر H-bridge بواسطة عناصر منفصلة، ولكن الأسهل والأرخص هو شراء دارة متكاملة لقيادة محرك التيار المستمر، وعلى سبيل المثال تعتبر الدارة المتكاملة LMD 18200 من شركة National semiconductor نموذجًا عن دارات قيادة المحركات وهي دارة متكاملة سهلة الاستخدام عالية التيار وتعمل بتيار حتى 3A وجهد يتراوح بين (12-55) فولت وهي أيضًا متآلفة مع عوائل(Cmos) ,(TTL) و تحوي ديودات تحديد داخلية وحماية من قصر الحمل وخرج مقاطعة للتحذير من ارتفاع درجة الحرارة، والدارة (L293D) هي أيضًا دارة (Ic) متكاملة شائعة الاستخدام في قيادة المحركات وهي رخيصة الثمن وسهلة الاستخدام ولكنها لا تعمل عند نفس تيارات الدارة (LMD 18200) ولا تمتلك أية ميزات إضافية .

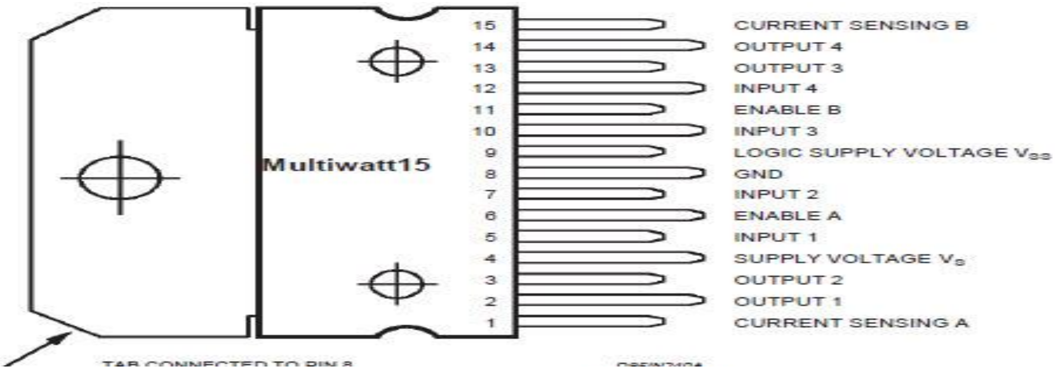
توجد أنواع أخرى عديدة من الدارات المتكاملة المخصصة لقيادة المحركات وغيرها من اللوحات المصنعة مسبقًا لقيادة عدد من المحركات .

فلقيادة المحركات المستمرة لعجلات المركبة استخدمنا شريحة قيادة محركين LS_298 DC ضمن موديول جاهز متلائم متناسب مع لوحة الأردينو



BLOCK DIAGRAM





يمكن أن تغذي حمل حتى ٥٠ فولط فعلياً ٤٦ فولط بالحياة التجريبية وبتيار $I_L = 2A$ جهد تغذية الدارة المنطقية من ٤.٥حتى ٧ فولط بتيار أقصاه ٧٠ميلي أمبير أما جهد قدح قواعد ترانزستورات منافذها ف٢٠ ميلي أمبير بجهد على الأقل ٣ فولط .

وكيفية التعامل معها بسيطة وتتضح من الرسومات السابقة في الصور .

٢.٣ ـ سيرفو التحكم عن بعد (RC)

سيرفو التحكم عن بعد RC (Remote Control Servo) هو أداة تشبه المحرك ويصمم لتطبيقات التحكم بالموضع التي تشبه التحكم بمؤشر (Pointer like position control).

تستخدم هذه الأدوات إشارة خارجية ذات عرض نبضة معدل PWM ،

للتحكم بموضع الجذع ضمن حيز صغير من مجال الدوران الأعظمي

ويتم تغيير عرض النبضة من أجل تغيير وضع الجذع

ويتحدد مقدار الدوران الزاوي لجذع السيرفو بحوالي 180 الى 210 درجة ((لكن هناك طرق معينة لتعديلها حتى ٣٦٠ درجة) ويتعلق ذلك بنوع السيرفو المستخدم ويستطيع السيرفو تأمين قدر مناسب من العزم منخفض السرعة (بسبب وجود نظام تخفيض سرعة) كما يؤمن سرعات فتح وإغلاق متوسطة من أجل إزاحة كاملة .يستخدم السيرفو عادة في الأجهزة الآلية (robotics) و للتحكم بالاتجاه في نماذج السيارات والقوارب والطائرات إضافة إلى استخدامه في تطبيقات أخرى.

يبدو سيرفو RC المعياري كصندوق بسيط له جذع قيادة وتخرج منه ثلاثة أسلاك وهذه الأسلاك هي سلك تغذية(عادة لونه احمر) وسلك أرضي (عادة لونه اسود) وسلك للتحكم بموضع الجذع (ويختلف لونه حسب الجهة الصانعة وهنا برتقالي للسيرفو SG90 الذي استخدمناه والذي يتحمل جذعه ما يقارب 0.5kg) يوجد في الصندوق محرك تيار مستمر، أداة تغذية عكسية، ودارة تحكم .

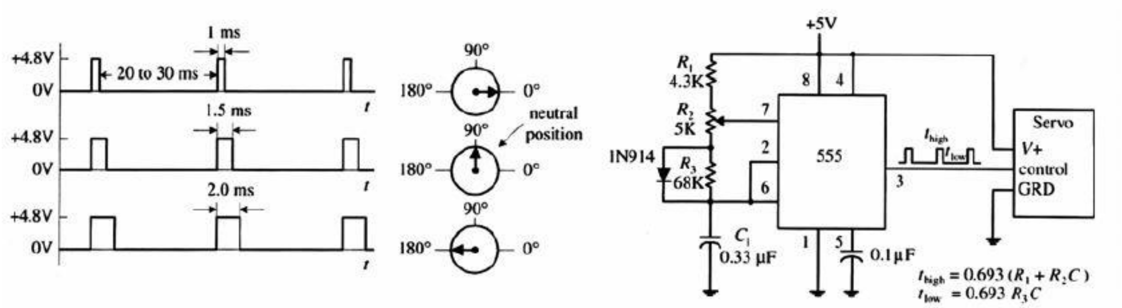
تتكون أداة التغذية العكسية عادة من مقسم جهد مقبض تحكمه موصول مع المحرك عن طريق سلسلة من مخفضات السرعة، عند دوران المحرك يدور مقبض التحكم بمقسم الجهد، وعادة تكون زاوية دوران الجذع محددة ب 180 او 210 درجات ، وبذلك لا يدور مقبض مقسم الجهد بشكل دائري كامل .

يستخدم مقسم الجهد كعنصر مراقبة للموضع وبواسطته تتعرف الدارة تمامًا على مقدار دوران الجذع (من خلال مقاومته) . تستخدم دارة التحكم قيمة مقاومة مقسم الجهد مع إشارة دخل بعرض نبضة معدل لقيادة المحرك بحيث يدور عددًا محددًا من الدرجات ثم يتوقف (يختلف مقدار

عزم التوقف من سيرفو إلى سيرفو) ويتحدد مقدار دوران جذع السيرفو بعرض نبضة إشارة الدخل. فالتحكم به فعلياً عن طريق تقنية pwm ولتبسيط المفهوم :

نرسل إشارة التحكم pwm ١٠٠ بتردد هرتز تكافئ ١٠ ميلي ثانية من اليويو بورد فيقوم السيرفو بترجمة نبضة pwm وتعديل زاوية دورانه وفقها :

فرضاً أرسلنا نبضة عرضها ١ميلي ثانية مفترضين بالحساب أن ٠ ميلي ثانية يكافئ ٠% و ١٠ ميلي ثانية تكافئ ١٠٠% فالزاوية التي يدورها ذراع السيرفو هي ١٨٠*١مقسومة على ١٠ أي هي بالنتيجة ١٨ درجة وهكذا إذ ١٠ ميلي ثانية تقابل دوران ١٨٠ درجة على اعتبار السيرفو ذو دوران حده الأعظمي ١٨٠ درجة كما في مشروعنا وللتوضيح أكثر من المخططات البيانية بالأسفل وإليكم إسهاباً مفصلاً عن طريقة التحكم بها أيضاً



الشكل(17)

عندما يكون عرض النبضة (1.5ms) يدور السيرفو إلى الموقع الحيادي

(مثلاً الزاوية 90 هي الموقع الحيادي بين الزاوية 180-0) ولكي يدور السيرفو عددًا محددًا من الدرجات من الموقع الحيادي يتم تغيير عرض إشارة التحكم، ولجعل السيرفو يدور بعكس عقارب الساعة تطبق على مدخل التحكم إشارة عرضها أكثر من 1.5ms ولتدوير الجذع مع عقارب الساعة تطبق على مدخل التحكم إشارة بعرض أقل من 1.5ms ويختلف عرض النبضة كي يدور السيرفو ويحقق انزياحا زاويًا دقيقًا من سيرفو إلى آخر فمثلا يحقق بعض أنواع أجهزة السيرفو دورانًا أعظمي بعكس عقارب الساعة إذا كان عرض النبضة 1ms ودورانًا أعظمي مع عقارب الساعة عند عرض نبضة يساوي 2ms أما بعضها الآخر فيحقق الدوران الأعظمي بعكس عقارب الساعة عند عرض نبضة 1.25ms وبعكس عقارب الساعة عندما يكون عرض النبضة 1.75ms يستخدم عادة جهد تغذية يساوي 4.8v لتغذية السيرفو وقد تصل قيمة هذا الجهد إلى 6v حسب نوع السيرفو المستخدم وطرازه .يختلف التيار الذي يستهلكه السيرفو كثيرًا حسب استطاعة خرج السيرفو .يمكن استخدام دارة 555 بسيطة كالدارة المبينة في الشكل لتوليد إشارة

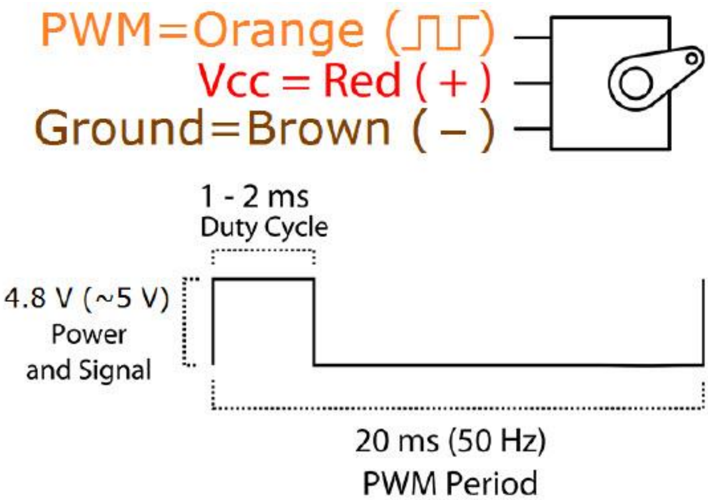
قيادة السيرفو وتستخدم R2 للتحكم بعرض النبضة .يمكن أيضًا قيادة السيرفو بواسطة معالج صغري (microprocessor) أو بواسطة متحكم صغري (Microcontroller) عند استخدام السيرفو المتحكم في نموذج طائرة ترسل إشارة تحكم أولية (يتم توليدها عن طريق تغيير قيمة مقسم الجهد المستخدم كمتحكم) الى دارة معدل موجة راديوية تقوم بترميز إشارة التحكم ضمن الموجة الحاملة ويتم بث الموجة الحاملة كموجة راديوية (radiowave) بواسطة الهوائي وترسل الموجة الراديوية إلى هوائي في الاستقبال في الطائرة ومنه إلى دارات المستقبل في جهاز التحكم حيث تتم استعادة إشارة التحكم عن طريق كشف التعديل وتطبق إشارة التحكم على السيرفو المخصص للتعامل معها وإذا كانت دارة التحكم تحوي أكثر من سيرفو فمن الضروري استخدام أكثر من قنال لإرسال إشارات التحكم.

تحتاج أغلب الطائرات المتحكم بها لا سلكيًا إلى جهاز لا سلكي بأربع أقنية للتحكم بالطائرة حيث تستخدم إحدى الأقنية للتحكم بالجنيحات وقناة أخرى للتحكم بالموجة (rudder) والثالثة للتحكم بالسرعة والرابعة للتحكم بالارتفاع، وتستخدم قناة خامسة أحيانًا للتحكم بميزات أخرى كالتحكم بآلية الهبوط . تخصص الأقنية (6-11) للمجال الترددي (72mhz) للتحكم بالطائرات ويمكن أيضًا استخدام مجال ال50mhz توجد ترددات في مجال الـ 72mhz مخصصة أيضًا لأغراض التحكم ، وإذا كنت مهتمًا بأجهزة السيرفو المقادة عن بعد (RC Servos)

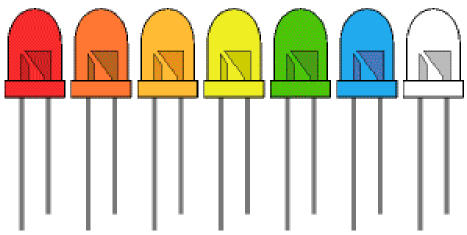
بواسطة الأجهزة اللاسلكية ننصحك بالبحث في المحلات التجارية التي تتعامل بهذه الأجهزة لأنها تقدم أجهزة إرسال واستقبال وأجهزة سيرفو لهذه الأغراض. يمكن تحويل السيرفو إلى محرك قيادة بدوران غير محدود عن طريق إلغاء حلقة التغذية العكسية وفصل الطرف الثالث لمقسم الجهد وعدم وصل (ربط) نظام تغيير السرعة (gear system) بحيث يدور المحرك (360) درجة ، يمكن استبدال مقسم الجهد بمقاومتين ثابتتين يمكن استبدال مقسم الجهد بمقاومتين ثابتتين (ويحل خرج مقسم الجهد محل الذراع الثالثة للمقاومة المتغيرة) يستخدم مقسم الجهد لجعل دارة التحكم بالسيرفو تضبط الجذع على الوضع الحيادي. يمكن تحديد قيم المقاومات الثابتة التي تستخدم بدل المقاومة المتغيرة بواسطة مقياس أوم وقياس المقاومات بين الذراع الثالث للمقاومة المتغيرة والأذرع الأساسية.

بعد تحويل السيرفو إلى محرك، يمكن تدوير المحرك باتجاه عقارب الساعة بتطبيق نبضة بعرض (1.5ms) على مدخل التحكم وطالما بقيت إشارة التحكم مطبقة على الدخل يستمر المحرك بالدوران- لأنك ألغيت نظام التغذية العكسية. من أجل تدوير السيرفو بعكس عقارب الساعة تطبق على الدخل نبضة بعرض أقل من (1.5ms)





Position "0" (1.5 ms pulse) is middle, "90" (~2 ms pulse) is all the way to the right, "-90" (~1 ms pulse) is all the way to the left.



٣.٢ عناصر الكترونية بسيطة :

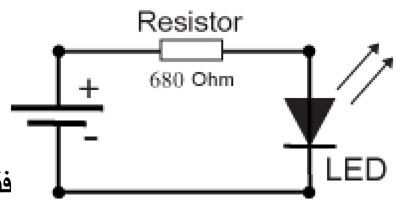
مقاومات لتحديد التيار و ٤ ليدات ضوئية



العلاقة التالية توضح قانون المقاومة في حالات وصلها :

$$R = \frac{V}{I}$$

Ohm's Law. R is Resistance, V is Volt, I is Current

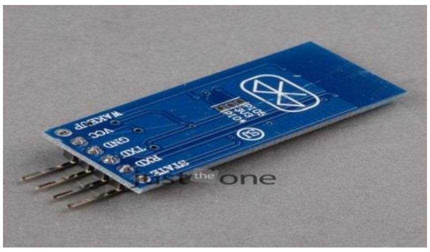


فقانون حماية الليد R= (Vs-Vf_max) / If وفق

Type	Colour	I_F max.	V_F typ.	V_F max.	V_R max.	Luminous intensity	Viewing angle	Wavelength
Standard	**Red**	**30mA**	**1.7V**	2.1V	5V	5mcd @ 10mA	60°	660nm
Standard	**Bright red**	**30mA**	**2.0V**	2.5V	5V	80mcd @ 10mA	60°	625nm
Standard	**Yellow**	**30mA**	**2.1V**	2.5V	5V	32mcd @ 10mA	60°	590nm
Standard	**Green**	**25mA**	**2.2V**	2.5V	5V	32mcd @ 10mA	60°	565nm
High intensity	**Blue**	**30mA**	**4.5V**	5.5V	5V	60mcd @ 20mA	50°	430nm
Super bright	**Red**	**30mA**	**1.85V**	2.5V	5V	500mcd @ 20mA	60°	660nm
Low current	**Red**	**30mA**	**1.7V**	2.0V	5V	5mcd @ 2mA	60°	625nm

الجدول:

٣.٣ شريحة موديول بلوتوث HC-06:



توصل مع الأردينو مع أقطاب الإرسال والاستقبال RX/TX لتبادل البيانات وإرسال الأوامر لتأمين الاتصال اللاسلكي مع جهاز الهاتف الذكي مجال بعدها بسيط بضعة أمتار فعلياً ٣ متر تغذيتها ٣.٣ فولط ولها قطب أرضي وقطبين إرسال واستقبال وقطب ليد حالة يضيء بثبات عند تحقق الاتصال ضمن المجال ويومض في حال انقطاعه وبالنسبة لتهيئتها ومعرفة عنوان قناع الاتصال لها فمن أوامر نافذة Commander

أو تطبيقات حديثة لرؤيته في أجهزة الموبايل وذلك من خلال الأوامر

(AT command is case- sensitive, should end up with terminator ("enter" or "\r\n").)

1. Test

Command	Response	Parameter
AT	OK	None

2. Reset

Command	Response	Parameter
AT+RESET	OK	None

3. Get the soft version

Command	Response	Parameter
AT+VERSION?	+VERSION: <Param> OK	Param: Version number

و طريقة الوصول نراها بالقسم العملي التنفيذي للمشروع .

أيضاً للاتصال اللاسلكي مع بورد اليويو التي تدعم بلوتوث الـUSB على منفذها استخدمنا بلوتوث دنغل USB والتي تعد أفضل لأنها تخولنا التحكم باليويو عن طريق الاتصال اللاسلكي عبرها لمسافة تصل ١٠٠ متر تقريباً .



توضيح مفهوم الوصل اللاسلكي البلوتوث

ما هو البلوتوث What is The BlueTooth؟

Bluetooth®

تعريف تقنية البلوتوث هي حقيقة عبارة عن معيار(أو طريقه) للاتصال اللاسلكي عبر موجات الراديو(**RF**) قصير المدى بين أجهزه تشكل شبكه شخصيه محدودة المسافة) (**PAN**) حوالي ١٠ أمتار..... وبالتالي أي جهازين يتبعوا نفس هذا المعيار يمكنهم الاتصال وتبادل البيانات فيما بينهم دون الحاجة إلى اتصال مباشر فيما بينهم...

كانت بداية ما يعرف الآن بتقنيات بلوتوث (**BlueTooth**) عندما بدأتها شركة أريكسون في عام ١٩٩٤ثم تبعتها شركات كثيرة في هذا التوجه (أكثر من ألف شركه) تحت ما يسمى (مجموعة الاهتمام الخاصة بالبلوتوث أو(**BlueTooth special interest group (SIG)**)، حيث كان الغرض من هذه التقنية أن تكون بديلاً فعالاً لأسلاك التوصيل بحيث يمكن ربط جهاز الكمبيوتر بجميع ملحقاته بطريقة لاسلكية (**Wireless**) فبدأت الشركات تتنافس في إنتاج الأجهزة والملحقات التي تدعم هذه التقنية.

في هذا التقرير سوف نتحدث عن خلفية تسمية البلوتوث ثم نذكر مزايا هذه التقنية والفرق بينها وبين تقنية الاتصال باستخدام الأشعة تحت الحمراء (**InfraRed**) ثم نتحدث عن نوعية الشبكة التي تدعمها التقنية

ما هي التطبيقات العملية للبلوتوثwhat're The BlueTooth Applications

إنها تفوق تخيلنا في الحقيقة لكن الطموح مستمر وعالي التفكير وعمليا جميع الأجهزه الرقمية الحديثة سواء في المكتب أو في البيت والتي لا تستخدم معيار اتصال متزامن فيما بينها ممكن أن تحقق التواصل تحت مظلة تكنولوجيا بلوتوث **BlueTooth**

من أمثلة هذه الجهزه : الهاتف وأجهزة النداء **pager** – الحاسب الألي - الطابعة – المفكرة الشخصية - أجهزة الفاكس – الحاسب الرقمي المحمول**laptop**

وغيرها الكثير.



سبب التسمية The Naming Reason

تعود التسمية إلى ملك الدانمارك هارولد بلوتوث **Harold BlueTooth** الذي وحد الدانمارك والنرويج. واختير هذا الاسم لهذه التقنية للدلالة على مدى اهتمام الشركات الدنمركية والنرويجية بالاتصالات اللاسلكية . وبالرغم من أن هذه التسمية لا علاقة لها بمضمون التقنية إلا أنها انتشرت وأصبحت معروفة على المستوى العالمي.

تحديات تقنية البلوتوث BlueTooth technical challenges

الوظائف السابق ذكرها تتضمن بعض التعقيدات ليس فقط لكثرة التطبيقات التي تدعمها البلوتوث ولكن أيضا بالنسبة إلى حجم الشريحة ومقدار القدرة التي تستهلكها لتنفيذ المطلوب منها بنفس الكفاءة والدقة إذا للنظام التقني الذي يدعم البلوتوث يجب أن يواجه التحديات التالية:

١) يجب عليه أن يستخدم مدى من التردد غير المرخص للإستخدام العام... لذا تم اختيار مدى تردد الأجهزة العلمية و التطبيقية **(ISM)** (**Industrial Scientific and Medical**) لكي يعمل فيه نظام البلوتوث.. التحدي هنا أن يواجه أي تشويش الصادر من مصادر أخرى (أجهزةأخرى) تعمل في هذا المدى

٢) أن يكون جهاز الإرسال والاستقبال الـ **transceiver** يجب أن يتوافق مع تغير البيئة المحيطة لأن الجهاز بطبيعةالأمر سيكون محمولاً

٣) يجب ألا يشغل مساحة كبيرة داخل الجهاز الحامل له وكذلك ألا يستهلك إلا مقدار قليل من القدرة **power** من الجهاز العائل

٤) يجب أن يدعم أكثر من نوع من الأجهزة في نفس الوقت

٥) أن يضمن التزامن بين الوحدات المتصلة بين الأجهزة المشاركة في نفس الوقت والتي تستخدم أنواع مختلفة من صيغ الاتصال فيما بينها

٦) الأمان الممكن أن يقدمه ال "بلوتوث" في نقل البيانات الشخصية والأوامر

مميزات تقنية البلوتوث BlueTooth Tech. Specifications

لكي نقتنع بجدوى استخدامنا لتقنية البلوتوث ومدى ملاءمتها لنا كأفراد ومؤسسات يجب علينا معرفة مميزات هذه التقنية والتي منها:

١) بديل فعال لأسلاك التوصيل، حيث يمكن لتقنية البلوتوث تغطية المسافات (من ١٠ إلى١٠٠متر) في جميع الاتجاهات، وذلك من خلال استخدام موجات الراديو (**RF**) قصيرة المدى بتردد (٢.٤٥جيجاهرتز) وهذا التردد يتفق مع الأجهزة الطبية والأجهزة العلمية والصناعية مما يجعل انتشار استخدامه سهل.

٢) يمكن استخدامها لتشكيل شبكة محلية بين الأجهزة التي تقع ضمن نطاق التغطية (**BlueTooth Zone**).

٣) رخيصة الثمن مقارنة بتقنيات الاتصالات اللاسلكية الأخرى حيث لا تتجاوز قيمة الشريحة (٥–٢٠) دولار.

٤) إمكانية نقل الأجهزة من مكان إلى آخر دون أي تغيير في الإعدادات.

٥) يستهلك قدره (**power**) قليل بمقارنته بغيره من تقنيات اللاسلكي.

٦) يجتنب أي تشويش مع غيره من الأجهزه المحموله (سيأتي شرحه فيما بعد)

٧) يمكن نقل البيانات و الأصوات عن طريقه.

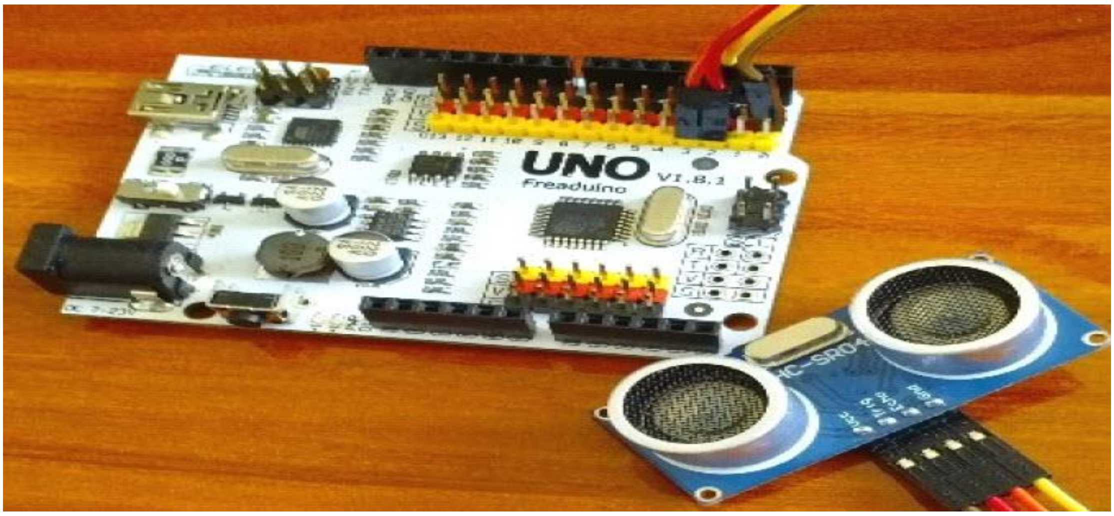
٨) سهلة التشغيل حيث تستطيع الأجهزة أن تتواصل مع بعضها البعض بدون تدخل المستخدم وكل ما عليك هو الضغط على زر التشغيل واترك الباقي للبلوتوث ليتحاور مع الجهاز المعني بالأمر من خلال ال (**Module**).

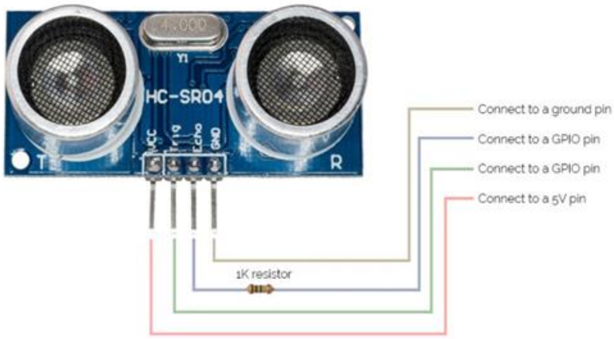
هل تتداخل الإشارات المتبادلة عن طريق البلوتوث Does The Bluetooth Reciprocal Signals Interfere؟

عند تواجد العديد من الأجهزة الالكترونية في الغرفة يمكن أن يحدث تداخل لأنني ذكرت أن مدى تأثير البلوتوث في حدود ١٠ متر وهو أكبر من مساحة الغرفة، مشكلة التداخل تم حلها بطريقة ذكية حيث أن إشارة البلوتوث ضعيفة وتبلغ ١ ميلي وات إذا ما قورنت بإشارات أجهزة الموبايل التي تصل إلى ٣ وات. هذا الضعف في الإشارة يجعل مدى تأثير إشارات البلوتوث في حدود دائرة قطرها ١٠ متر ويمكن لهذه الإشارات اختراق الجدران مما يجعل التحكم في الأجهزة يتم من غرفة لأخرى دون الحاجة للتواجد مقابل الجهاز المعني مثلاً، ولكن هذا الاحتمال غير وارد لأن هناك مسح متواصل لمدى ترددات إشارة البلوتوث، وهذا ما يعرف باسم **Spread-Spectrum Frequency Hopping** حيث أن المدى المخصص لترددات البلوتوث هي بين ٢.٤٠ إلى ٢.٤٨ جيجا هيرتز ويتم هذا المسح بمعدل ١٦٠٠ مرة في الثانية الواحدة. وهذا ما يجعل الجهاز المرسل يستخدم تردد معين مثل ٢.٤١ جيجا هيرتز لتبادل المعلومات مع جهاز أخر في حين أن جهازين في نفس الغرفة يمكن أن يستخدموا تردداً آخر مثل ٢.٤٤ جيجا هيرتز ويتم اختيار هذه الترددات تلقائيا وبطريقة عشوائية مما يمنع

حدوث تداخلات بين الأجهزة، لأنه لا يوجد أكثر من جهازين يستخدمان نفس التردد في نفس الوقت. وإن حدث ذلك فإنه يكون لجزء من الثانية.

٣.٤ حساس المسافة HC-SR04 UltraSonic:





الصوت وانتشاره

إن تبادل المعلومات بالأمواج الصوتية (المسموعة) وسيلة عرفتها الكائنات الحية منذ بدء الخليقة، واستفادت منها في تطوير أساليب الحياة، والتأقلم مع المحيط الخارجي وخاصة الإنسان، حيث قام بتطوير عدة وسائل اتصالات مسموعة.

توليد الصوت:

أقرب مثال لأداة توليد أو احداث الصوت هي الحبال الصوتية للإنسان . عندما يتحدث الإنسان يقوم الهواء الخارج من الرئة بإحداث ذبذبات أو اهتزازات مكيانيكية في الحبال الصوتية وتنتقل هذه الذبذبات إلى جزيئات الهواء الخارجي ، وبما أن جزيئات الهواء مشدودة إلى بعضها بروابط قابلة للتمدد فإن ذلك يساعد على انتقال هذه الاهتزازات من جزيء لآخر على شكل ترددات مضغوطة مبتعدة عن مصدر الصوت بسرعة 330 m/s متر بالثانية وهي سرعة الصوت في الهواء.

استقبال الصوت:

مبدأ استقبال الصوت يمكن توضيحه عن طريق حاسة السمع حيث تعمل طبلة الأذن كأداة استقبال تستقبل موجات الصوت (الذبذبات المكيانيكية) وتحولها إلى إشارات كهربائية تقوم بإرسالها إلى الدماغ الذي يقوم بتحليل هذه الإشارات .

والصوت المسموع لدى الإنسان يتراوح بين (20 و 20000 هزة في الثانية ،وهو ما يدعى بالتردد Hz) .

ويمكن اصدار الأمواج الصوتية المسموعة بعدة طرق مثل الجرس والآلات الموسيقية أو الحث الكهربائي للمجال المغناطيسي كما في مكبرات الصوت.

انتقال وانعكاس الصوت:

إن انتقال الصوت في مادة ما يتأثر بتكوينها ،فسرعة انتقال الصوت في المادة تتأثر أساساً بكثافة المادة وقوة الروابط بين جزيئاتها، وعند اصطدام الموجة الصوتية بعائق (الجدار مثلاً) فإنها ستنعكس وتنتقل جزئياً ، أين أن جزء من الصوت سينتقل في الجدار بسرعة تحدد مقدارها نوعية المادة المكونة للجدار بينما ينعكس جزء آخر من الصوت تلتقطه الأذن مرة أخرى ،

(مثل صدى الصوت في المناطق الجبلية).

أقسام الصوت:

يتألف الصوت من ثلاثة أقسام هي:

- **أمواج مسموعة صوتية وتبلغ تردداتها بين (20-20000) Hz هرتز.**
- **أمواج تحت السمعية وتبلغ تردداتها أقل من 20 Hz هرتز.**
- **أمواج فوق الصوتية (السمعية) وتكون تردداتها أكثر من 20000 Hz هرتز.**

ولا يمكن للأذن البشرية التقاط الأمواج فوق الصوتية والأمواج تحت الصوتية.

بينما لبعض الحيوانات المقدرة على إرسال واستقبال الأمواج فوق الصوتية منها حيوان الدولفين البحري ، والحيتان ،والوطواط الذي يستعمل الذبذبات فوق السمعية لتحديد مساره واصطياد الحشرات في الظلام .

ويمكن توليد الأمواج فوق الصوتية ميكانيكياً كالصافرات أو بطريقة شحذ المعادن.

يوجد نوعين من الأمواج فوق الصوتية المستخدمة حالياً:

- **الموجات فوق الصوتية عالية الطاقة وتبلغ طاقتها من 1 إلى 100 واط وتستعمل في:**

  أ- نظافة جميع أنواع القطع المعدنية.

  ب- تسخين ولحام المواد مثل البلاستيك والرقائق المعدنية.

- **الموجات فوق الصوتية منخفضة الطاقة وتبلغ طاقتها من 1 ميلي واط إلى 1 واط وتستعمل في:**

  أ- تقنية السونار(SONAR) لقياس المسافة كقياس أعماق البحار وكذلك للأغراض العسكرية.

  ب-التحاليل الطبية

  ج-أجهزة التنبيه والعدادات.

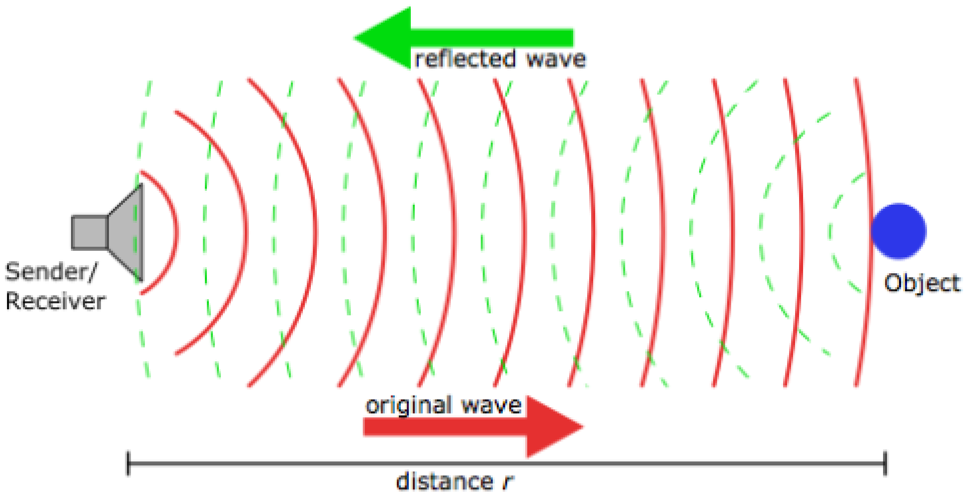
  د- الإختبار غير الإتلافي للمواد.

وسنركز في دراستنا على الحساس فوق صوتي الذي يعمل بتقنية السونار(SONAR) قياس المسافة.

والذي يمثل محاكاة لطائر الوطواط في التعرف على الحواجز والأجسام .

حيث يقوم الوطواط بإرسال موجات فوق صوتية واستقبال ارتدادات هذه الأمواج ،حيث يقوم تحليل المدة الزمنية بين الإرسال والإستقبال في دماغه وقياس المسافة بينه وبين الحاجز أو الهدف وبالتالي يتعرف على الأجسام التي تحيط به .

وهذه يمثل نظام رؤية آخر يعمل في الظلام .



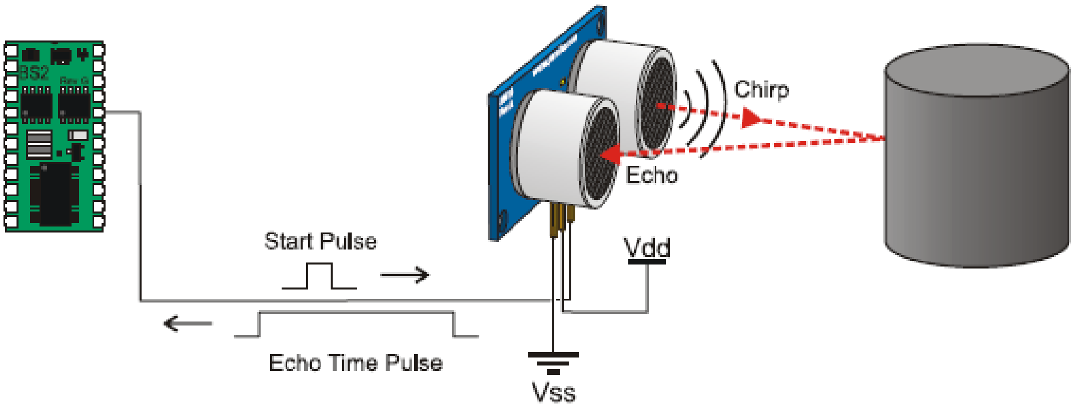
عملية إرسال واستقبال الأمواج فوق الصوتية ودورها في تحديد المسافة

فحساس المسافة هو جهاز نستطيع استخدامه لقياس بعد جسم ما عن الحساس بمجال يتراوح بين 3cm-3.3m.

ويستخدم في تطبيقات الروبوت ومشاريع الأتمتة ، حيث يقيس مسافة الأجسام بدقة تصل إلى نصف سنتميتر.

آلية عمل The Ping))) ultrasonic Sensor

الشكل (2) يظهر كيف يصدر الحساس ذبذبات عن طريق مكبر الصوت (فوق الصوتي) وبقياس الزمن حتى ارتداد الصدى للميكرفون (فوق الصوتي) .تبدأ عملية القياس بإرسال نبضة للحساس ثم ينتظر الحساس حتى ارتداد الصدى المنعكس من سطح الجسم حيث يعطي الحساس خرج TTL 5V طوال فترة الانتظار وفور اكتشاف الصدى في المستقبل يتحول خرج الحساس إلى 0V حيث يقوم البرنامج بتخزين الفترة التي بقي فيها خرج الحساس إشارة مرتفعة (5V) في متحول ليتم حساب المسافة بالإعتماد على سرعة انتشار الصوت في الهواء عن طريق البرنامج ويمكن حساب المسافة بعدة واحدات تبعاً للبرنامج (سنتميتر, انش, قدم,......)



The Ping))) Sensor آلية عمل

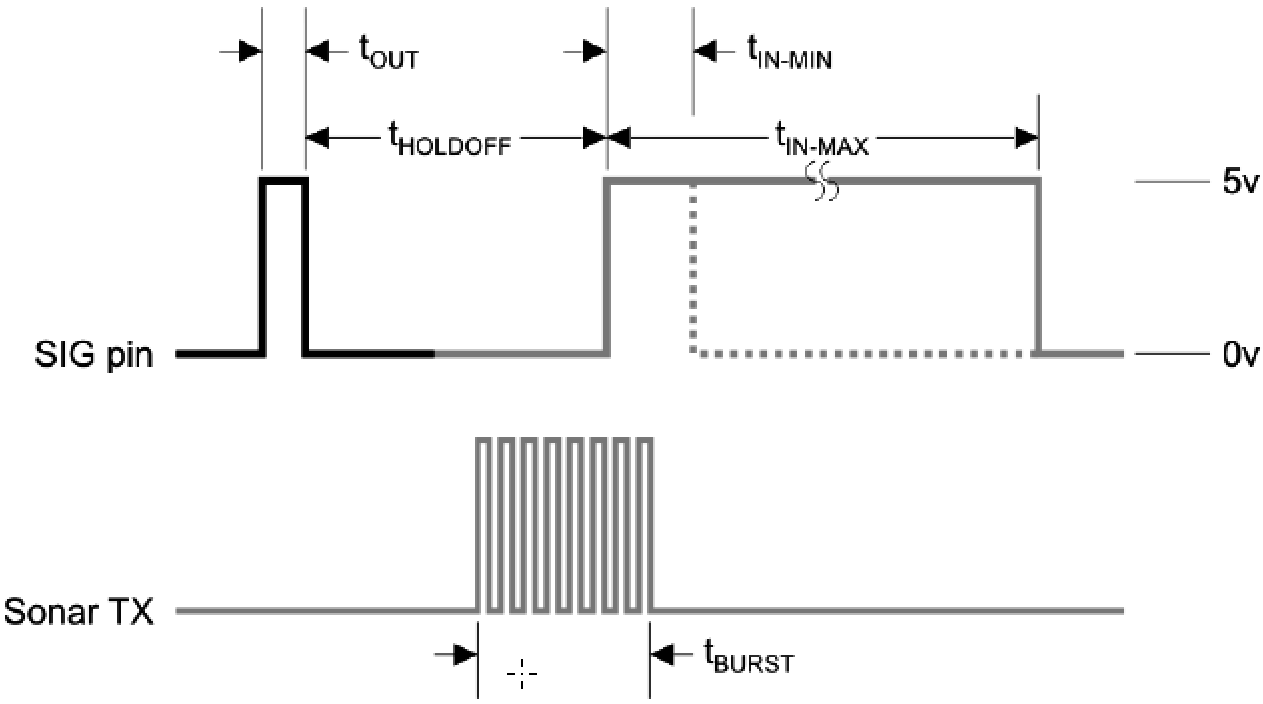
بروتوكول الإتصال:

يكشف حساس (((ping الأجسام عبر بعث انفجار قصير للأمواج فوق الصوتية ذات التردد 40KH ثم يدخل مرحلة الاستماع تحت تحكم المتحكم المصغر المضيف (نبضة القدح).

هذا الإنفجار ينتشر عبر الهواء ويصطدم بأي جسم يصادفه مرتداً عنه عائداً لمستقبل الحساس.

حساس (((ping يزود نبضة خرج للمتحكم المضيف والتي سوف تنتهي بمجرد اكتشاف الصدى.

وبالتالي يمكن قياس مسافة الهدف عبر تحليل عرض النبضة التي حصلنا عليها.



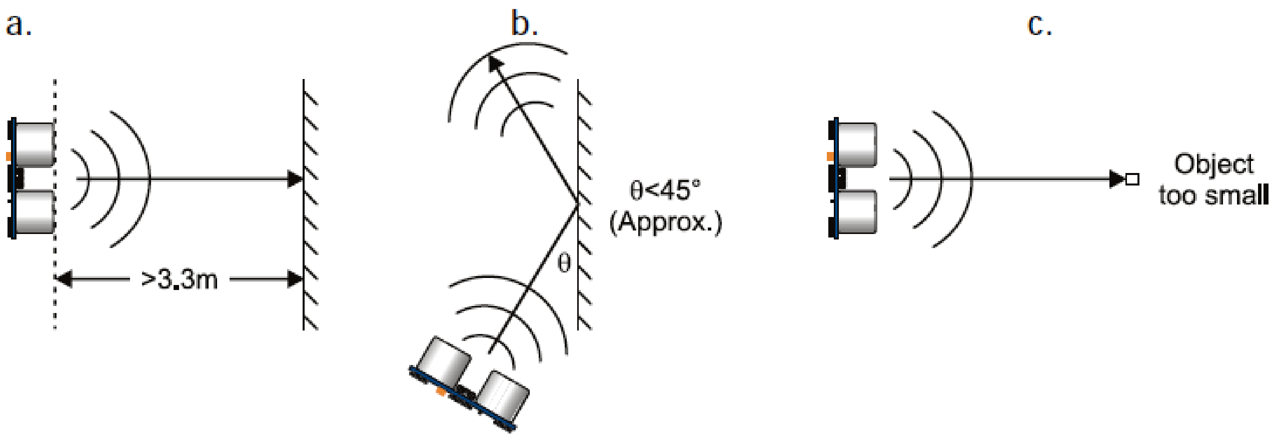
▬	Host Device	Input Trigger Pulse	t_{OUT}	2 µs (min), 5 µs typical
▬	PING))) Sensor	Echo Holdoff	$t_{HOLDOFF}$	750 µs
		Burst Frequency	t_{BURST}	200 µs @ 40 kHz
		Echo Return Pulse Minimum	t_{IN-MIN}	115 µs
		Echo Return Pulse Maximum	t_{IN-MAX}	18.5 ms
		Delay before next measurement		200 µs

الاستخدام العملي للحساس

1- وضعية الجسم:

لا يمكن لحساس (((PING قياس المسافة بدقة في الحالات التالية:

- بعد الجسم أكثر من 3 أمتار.
- زاوية الانعكاس عن السطح صغير بحيث لا يرتد صدى الأمواج الصوتية للحساس.
- إذا كان الجسم صغير جدا بحيث لا يمكنه عكس الأمواج فوق الصوتية.



الحالات التي تعطي قياس غير دقيق لحساس (((PING

2- نوعية المادة الهدف:

الأجسام التي تمتص الصوت أو التي تملك سطحاً شاذاً ربما لا تعكس الأمواج الصوتية بشكل كافي بحيث يتم اكتشاف صداها .

حساس (((PING يمكنه اكتشاف سطح الماء، لكنه غير معد للعمل في الهواء الطلق أو للإستعمال المستمر في البيئة الرطبة حيث يمكن أن يؤثر على الداء وعمر الجهاز.

3- درجة حرارة الهواء:

درجة الحرارة لها تأثير على سرعة انتشار الصوت في الهواء وبالتالي على عملية القياس وفق العلاقة:

$$C_{air}=331.5+(0.6*Tc)\ m/s$$

C_{air}: سرعة انتشار الصوت　　　　Tc:درجة حرارة الوسط

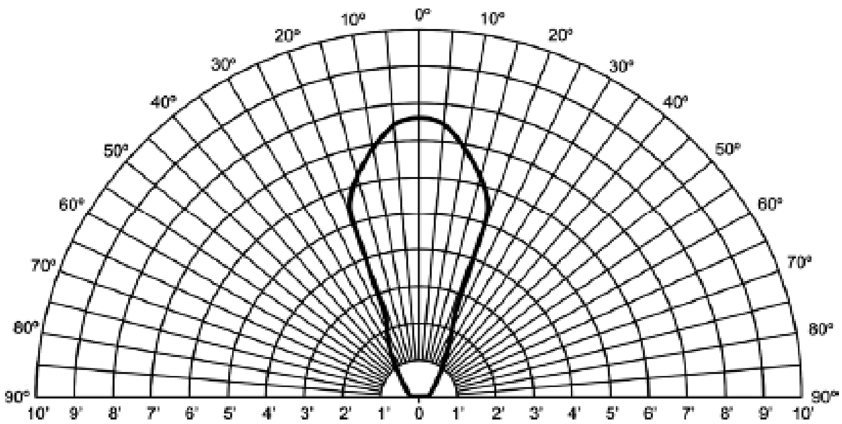
اختبار الحساس فوق الصوتي

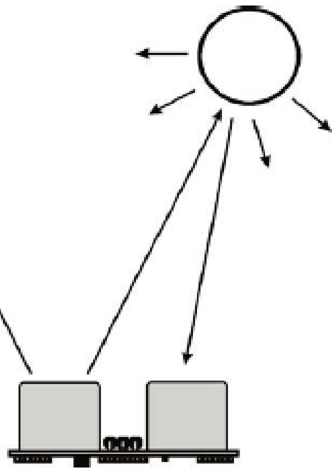
هذه الاختبارات أجريت على الحساس PING))) حيث تم اجراءها في مختبرات الشركة المصنعة Parallax

الإختبار الأول

ارتفاع الحساس : 40 in. (101.6 cm).

قطر الإسطوانة 3.5 in. (8.9 cm), الطول 4 ft. (121.9 cm) توجيه عمودي.



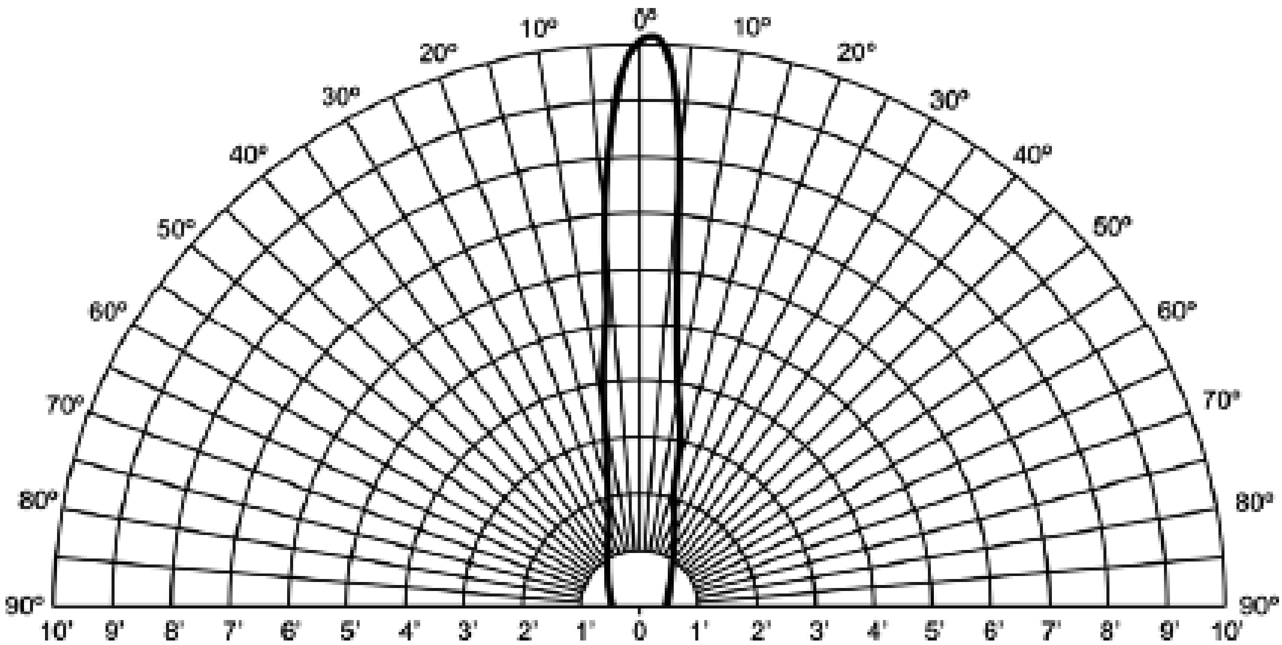


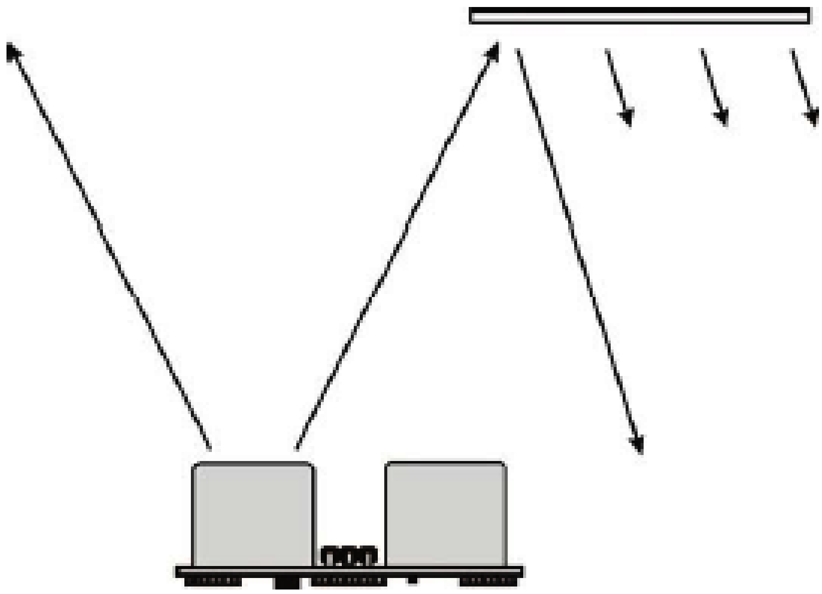
عندما تصطدم الموجة الفوق صوتية بحاجز دائري فإنها تنتشر بشكل عمودي على السطح الدائري وبالتالي فإن زاوية رؤية الحساس تكبر وبالتالي يمكن اكتشاف الجسم بسهولة أكبر كما يوضح الشكل.

الإختبار الثانى

ارتفاع الحساس : 40 in. (101.6 cm).

الهدف: لوحة أبعادها 12 in. x 12 in (30.5 cm x 30.5 cm) .





عندما تصطدم الموجة الفوق صوتية بحاجز مستقيم فإنها تنتشر بزاوية انعكاس تساوي زاوية الورود وتعاكسها بالنسبة لمحور الحاجز لذا فإن زاوية رؤية الحساس هنا صغيرة وبالتالي يصعب كشف الجسم كلما ابتعد الحاجز عن الحساس بشكل أفقي.

المميزات:

- مجال القياس يتراوح بين 2cm -3m (0.8 in to 3.3 yd) .
- كاشف ضوئي يوضح نشاط الحساس.
- خط مزدوج لنقل نبضات TTL في قطب I/O وحيد يستطيع الإتصال بواسطة 5V TTL أو 3.3 V CMOS.

- القدح يتم بواسطة نبضة TTL موجبة تتراوح مدتها بين (2 µs min, 5 µs typ).
- نبضة الصدى هي نبضة TTL موجبة تتراوح بين (115 µs min to 18.5 ms max).

HC-SR04

المواصفات:

- جهد التغذية: 5V dc.
- تيار التغذية: 30 mA typ; 35 mA max.
- الإتصال: نبضة TTL موجبة.
- درجة حرارة العمل الطبيعي: C (0 – 70°).
- الوزن: 9غرام.

$$L = C X T$$

-
- **الأقطاب**

GND	**Ground (Vss)**
5 V	**5 VDC (Vdd)**
trig	**Signal (I/O pin)** trigger
Echo	**Signal pulse recive**

بالنسبة لدارة توصيله بالأردينو سيتم توضيحها بإذن الله في القسم العملي التنفيذي .

٣.٥ بطارية لتغذية متحكمي الشريحتين يويو وأردينو بقيمة 9V:

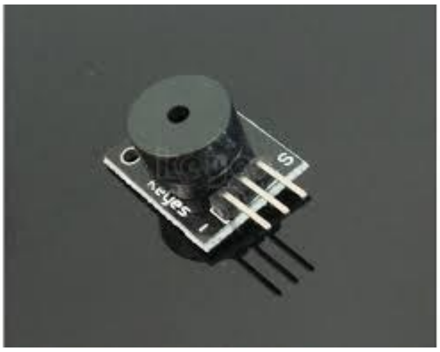


٣.٦ بطارية لتغذية شريحة القيادة والمحركات 12V:

حيث هي عبارة عن بطاريات ليثيوم خلايا لبطارية الحاسب المحمول LapTop تمدنا بطاقة مستمرة بجهد قيمته 12v وتيار ممتاز 2800mA



٣.٧ مكبر صوت صغير :



٣.٨ جهاز موبايل ذكي يعمل بنظام أندرويد:



الفصل الثالث

١.١ استعراض أفضل الطرق للتحكم عن بعد باستخدام الهواتف الذكية :

إنَّ أي تحكم يستدعي وجوباً وجود عقل يراقب ويحلل ثمَّ يتخذ القرارات ويعطي الأوامر بناءً على ذلك .

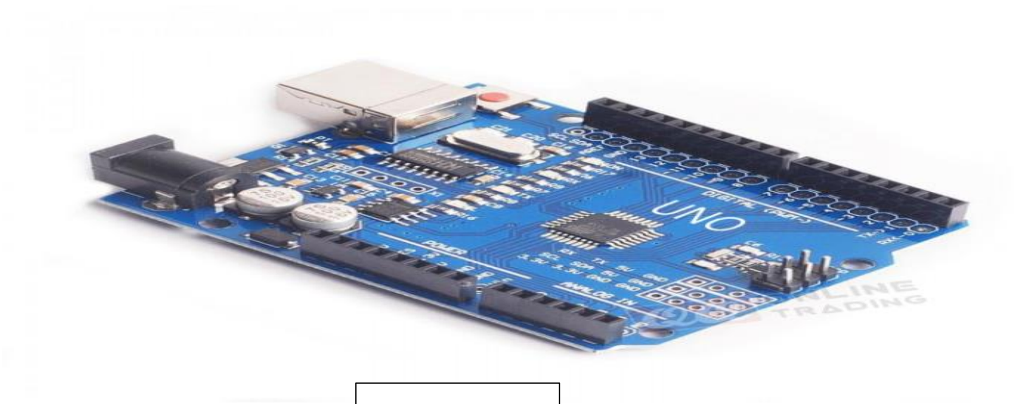
إذاً فإنَّ أي تحكم مهما كان تعقيده هو بالنهاية يعتمد على المعالجات والمتحكمات بدءاً من المايكرو وصولاً حالياً إلى شرائح التطوير

Arduino uno R3	Rasspbery PI MODEL B	IOIO/OTG

وإليكم مقارنة بسيطة بينهم لرؤية الأنسب بينها حسب المشروع والتطبيق قبل توضيح آليات وطرق ممكنة مستخدمة للتحكم عن بعد عن طريق أنظمة الهواتف الذكية :

المقارنة :

اسم اللوحة أو الشريحة	Arduino uno R3	IOIO/OTG	Rasspbery PI MODEL B
نوع المتحكم	لوحة الكترونية بمتحكم دقيق Microcontroller Atmega328	لوحة الكترونية بمتحكم دقيق PIC24FJ256DA210	حاسوب مصغر بمعالج Arm oc Arm 11
غاية التصميم	مصممة لتطبيقات التحكم الالكتروني فقط	مصممة لتطبيقات التحكم الالكتروني فقط بتخاطب مباشر مع تطبيقات الجافا	مصممة لتطبيقات البرمجة وقواعد البيانات ونظم تشغيل الحواسيب
لغة البرمجة	الـ C التقليدية وArduinoC	JAVA	تقبل كل لغات البرمجة التي يدعمها نظام لينكس
عدد منافذ بروتوكولات التخاطب والاتصال i2c ,spi ,UART	تحوي منفذين i2c و منفذ وحيد لكل من Spi , UART على حدا	أكثر من ١٥ منفذ	منفذ وحيد لكل من بروتوكولات الاتصال i2c UART, spi, على حدا
عدد المداخل التشابهية	٦	١٦ مدخل	لا يوجد
عدد المخارج الرقمية	١٤	٤٨ مخرج GPIO	٨ GPIO
عدد مخارج pwm	٦	٢٥مخرج	لا يوجد
أقل تيار تعمل عليه الشريحة	40MA	100MA	320-480MA
توصيل أجهزة إضافية عبر usb	لا يوجد	نعم يوجد منفذ	نعم يوجد منفذين
مدخل شبكة ايثرنت	لايوجد	لا يوجد	يوجد
جهد تشغيل اللوحة	5-17V	5-15V	5V
إمكانية التخاطب مع لغات برمجة أخرى	JAVA ,C++	لغة الجافا فيها تدعم إرسال أوامرها بصورة بايت وبإمكان أي لغة تفسير هذه الأوامر	ممكن
السعر	يبدأ من ٢٤$	تبدأ من ٣٥$	تبدأ من ٢٥$
إمكانية التخاطب المباشر مع نظام الأندرويد دون ملائمة برمجية	لا يمكن	يمكن	لا يمكن
سرعة المعالجة	جيدة	جيدة جداً	ممتازة
حجم أبعادها	متوسط معقول	صغيرة معقولة	متوسط معقول
وجود محاكاة افتراضية ببروتوس	يوجد	لا يوجد	لايوجد



VS

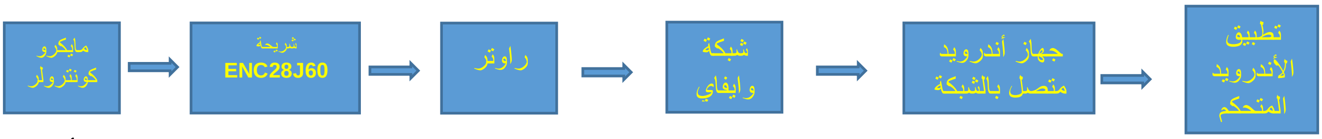


VS

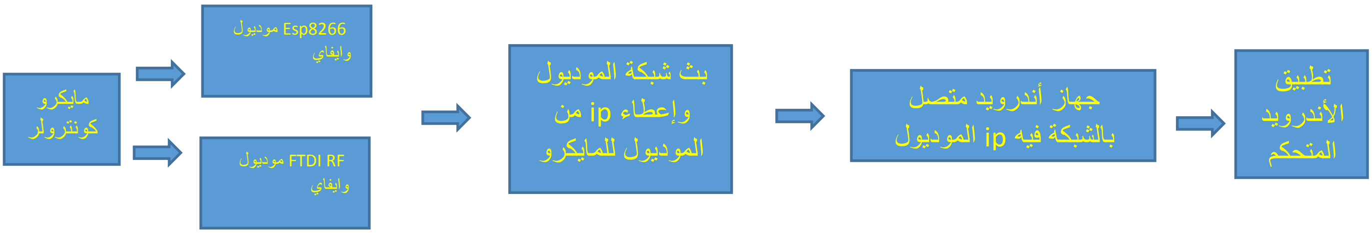


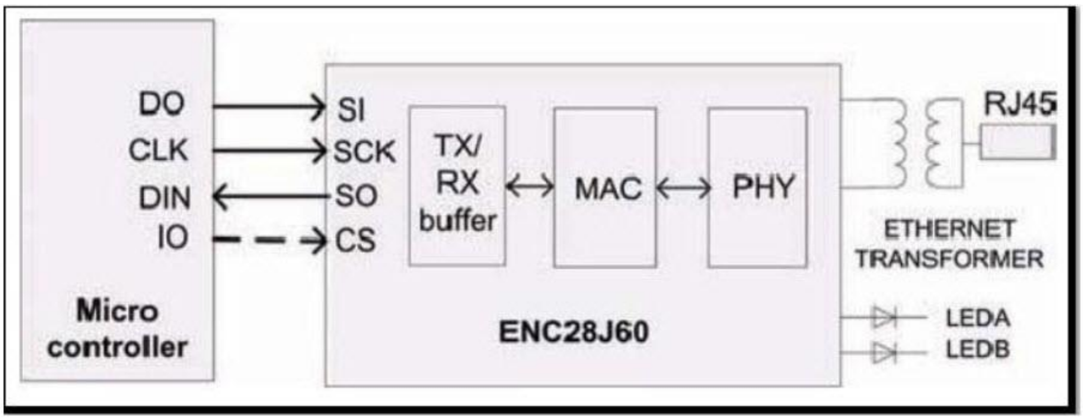
الطرق المشهورة والمستخدمة :

١) عن طريق المتحكم الدقيق مايكروكونترولر ولذلك عدة وسائل نذكر أشهرها :



أو





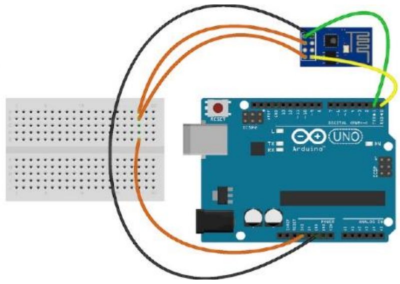
مخطط صندوقي لربط MCU بـ ENC28J60

حيث يتم التخاطب بهذه الطريقة والبرمجة عبارة عن برمجة المايكرو بلغته وبرمجة تطبيق الأندرويد بلغة الجافا ووضع عناصر ورموز محرفية برمجية معتمدة شيفرة أسكي ASCII للتخاطب بينهما .

٢) عن طريق لوحة الأردوينو :

ويتم ذلك بعدة طرق منها شبيهة الطريقة للمايكرو بموديولات أردوينو الخاصة كموديول Arduino Ethrnet Enc28j60 وموديول 8266





وهناك طريقتان أيضاً هما :

١*- استخدام شرائح ZigBee, Xbee مرسل مستقبل ذات المدى الواسع بالـKM .

٢*- استخدام موديول بلوتوث HC_06/HC_05 كالذي استخدمناه في مشروعنا لتحكم بالأردينو .

بكل ما سبق يتم التخاطب اعتماداً على شيفرة أسكي ASCII

٣) عن طريق بوردات وشرائح غوغل : وهي شرائح متطورة للخبراء .

٤) عن طريق الراسبيري باي المتصلة بشبكة راوتر والمتصل بذات الشبكة جهاز تطبيق الأندرويد المتحكم .

٥) عن طريق بورد اليويو:

وذلك بعدة طرق مميزة :

١*- جهاز أندرويد هدف فيه تطبيق التحكم الهدف متصل ببورد اليويو وصلاً مباشر عبر وصلة USb debugging وصلة الهواتف المحمولة السلكية وهو متصل بشبكة الراوتر نضع فيه IP الجهاز المتحكم وهناك جهاز أندرويد Boss متحكم فيه تطبيق التحكم ومتصل بشبكة الراوتر نضع فيه IP الجهاز الهدف .

٢*- شريحة FTDI_USB متصلة بمنفذ الـ USB في بورد اليويو من جهة ومن جهة أخرى متصلة أقطاب RX/TX فيها بموديول الوايفاي ESp8266 بالتالي يصبح لشريحة ioio عنوان IP خاص بها يبث بشبكة الموديول نضعه ضمن تطبيق الأندرويد للجهاز بعد تعريف سماحية الانترنت في كود اليويو وبالتالي إمكانية التحكم مباشرة وبسرعة عالية ومدى كبير .

٣*- شريحة اليويو متصلة مباشرة عبر منفذها الـUSB ب USB BLUETOOTH DONGLE وبالتالي إمكانية تطبيق الأندرويد المتضمن سماحية البلوتوث في كوده البرمجي أن يتحكم عن طريق الجهاز الذكي مباشرة باليويو وبسرعة لا بأس بها وبمدى يصل كأقصى حد لـ100m مع نسبة ارتياب . ((وهي الطريقة التي اخترناها للتحكم ببورد اليويو لتشغيل ذراع السيرفو)) .

هنا يتم التخاطب بين كود الجافا البرمجي مباشرة ً مع الشريحة .

وبما أنَّ التحكم عن بعد نحب أن نعرض نبذة مهمة مبسطة عن الشبكات اللاسلكية وبروتوكولاتها .

١.٢ شبكات الاتصال :

تعريف الشبكات اللاسلكية Wireless Networks

هي الشبكات التي تسمح للمستخدمين بالاتصال بالشبكة واستخدام تطبيقاتها بدون استخدام الأسلاك أو التي تعمل على نقل المعلومات بين العقد من دون استخدام الأسلاك.

إن هذا النوع من الشبكات ينفذ عادة مع نظم نقل معلومات بالتحكم عن بعد من خلال استخدام أمواج كهرومغناطيسية كالأمواج الراديوية كحامل لإشارة المعلومات.



أنواع الشبكات اللاسلكية

شبكات المناطق الشخصية Wireless Personal Area Network (PAN)

هي الشبكات التي تصل بين أجهزة ضمن مساحة صغيرة نسبياً ، عادة ما تكون هذه المساحة ضمن مجال يمكن لشخص الوصول إلى جميع أجزائه. كمثال على ذلك فإن تقنية Bluetooth تقوم مثلا بربط حاسوب شخصي مع سماعات.

شبكات المناطق المحلية Wireless Local Area Network (WLAN)

هي النوع الأكثر شيوعا من الشبكات اللاسلكية. تقوم بربط الأجهزة على مسافة أبعد من النوع السابق كمنزل أو مكتب أو حتى بناء وفي بعض الأحيان تمتد لتغطي عدة كيلومترات.

معظم الشبكات LAN تعتمد على المعيار IEEE 802.11 الذي يحتوي على معايير للشبكات اللاسلكية المحلية التي تعمل في الحزم الترددية 2.4\3.6\5 GHz وتضم عدداً من البروتوكولات المختلفة.

شبكات المناطق الكبيرة Metropolitan Area Network (MAN)

تقوم بربط عدة شبكات LAN مع بعضها البعض لتحقيق شبكة لاسلكية تمتد على رقعة جغرافية متوسطة الحجم مثال: عبر حرم جامعي أو مدينة ، الخدمة التي تؤديها مشابه للخدمة التي يقوم بها مزود الإنترنت Internet Service Provider (ISP).

شبكات الأجهزة الخلوية

إن التطور الذي حصل في الآونة الأخيرة في مجال الشبكات الخليوية مكننا من نقل معطيات ومعلومات عن طريق هذه الشبكات بالإضافة إلى الهدف الأساسي منها ألا و هو نقل المحادثات بين جهازين خلويين.

ويمكن تقسيم شبكات الأجهزة الخليوية الى ثلاثة اقسام حسب تطورها

- Global System for Mobile (GSM)
- The third generation of mobile telecommunications technology (3G)
- Long-Term evolution (4G LTE)

يمكن ربط دارة IOIO و ARDUINO بجهاز التحكم عن طريق استخدام أي نوع من أنواع الشبكات اللاسلكية السابقة..

قبل البدء يجب أن نتعرف على بعض المصطلحات.

بروتوكول الشبكة Protocol هو عبارة عن مجوعة من البرامج تقوم بتطبيق قواعد وطرق معينة في العمل فيما بينها لتمين الاتصال بين الأجهزة في الشبكة الواحدة ولكل شبكة بروتوكول خاص فيها و هذا لا يعني أن لكل شبكة بروتوكول فريد من نوعه خاص بشبكة معينة لا يوجد أي شبكة في العالم تستخدم هذا البروتوكول ولكن يوجد عدة بروتوكولات وتستخدم كل شبكة ما يناسبها منها.

إنترنت بروتوكول (Internet Protocol (IP بروتوكول يقدم اثنان من أهم وظائف مكدس بروتوكول الـ TCP/IP وهما العنونة والتوجيه. وهو عبارة عن العنوان التعريفي للجهاز الموصول على الشبكة

بوابة العبور الرئيسية Default Gateway هو عبارة عن عنوان الشبكة نفسها وهو أيضا بوابة العبور للأجهزة المربوطة على الشبكة لتتصل مع الشبكات أو الأجهزة المربوطة على شبكات أخرى.

المكررRepeater يستخدم لتقوية الاشارة عند إرسال البيانات بين الشبكات بشرط استخدام نفس البروتوكول.

الموجه Router يستخدم لربط الشبكات الفرعية مع بعضها البعض.

نقطة الوصول Access Point (AP) هو عبارة عن جهاز يقوم بأخذ كيبل إيثرنت (وتعني Lanأو شبكة محلية وكلها مسميات لنفس الشيء) من الشبكة المحلية أو من أي جهاز يراد ربطه بجهاز آخر وتحويل البيانات المراد تبادلها بين الجهازين إلى موجات راديو ويعمل على إرسالها واستقبالها ويعمل على إرسال البيانات إلى مسافات طويلة قد تصل إلى مئات الكيلومترات حسب قدرة وكفاءة الأجهزة المستخدمة والغرض الأساسي من هذا الجهاز إرسال واستقبال البيانات لمسافة طويلة ويمكن استخدامه كجهاز إرسال او استقبال.

```
C:\>ipconfig

Windows IP Configuration


Wireless LAN adapter Wireless Network Connection 3:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Wireless Network Connection 2:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Wireless Network Connection:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Ethernet adapter Local Area Connection:

   Connection-specific DNS Suffix  . :
   IPv4 Address. . . . . . . . . . . : 192.168.200.46
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.200.1
```

شرح كيفية إنشاء شبكة على الراوتر والاستفادة منها بالتحكم:

أولا نقوم بإنشاء شبكة لاسلكية على الموجة اللاسلكي Wireless Router ونقوم بإعطاء اسم للشبكة اللاسلكية Service Set Identifier (SSID) ونقوم بحماية هذه الشبكة باستخدام كلمة عبور Password تسمح للمستخدمين بالارتباط بهذه الشبكة. وتكون عنوان الشبكة هو نفسه بوابة العبور Default Gateway للمستخدمين المتصلين بهذه الشبكة.

نقوم بالاتصال من الكمبيوتراللوحي وجهاز الأندرويد للشبكة اللاسلكية وعند إتمام عملية الاتصال وإدخال كلمة العبور سوف يأخذ كلا من الجهازين عنوانه الخاص على هذه الشبكة (IP).

يمكننا معرفة عنوان جهاز التحكم عن طريقة نافذة التعليمات CMD بواسطة تعليمة IPconfig. نقوم بأخذ ال IP الخاص بالكمبيوتراللوحي ووضعة في برنامج الاستقبال الخاص بدارة IOIO على جهاز الأندرويد.

١.٣ شرح عن المشروع

جاءت فكرة المشروع لتظهر امكانية الاجهزة الذكية في التواصل مع المتحكمات عن طريق بعض وسائل الاتصال مثل Wi-Fi او Bluetooth او حتى Internet

لقد اخترنا في مشروعنا تقنية Bluetooth الاقل تكلفة نسبيا مقارنا مع وسائل الاتصال الأخرى والمتوفرة بالأسواق حالياً .

كانت فكرة التحكم بالعناصر الأساسية أو الحيوية لمعظم الأجهزة الالكترونية كالمحركات و الليدات و مكبرات الصوت لأخذ فكرة عامة عن مدى امكانية التحكم بأي جهاز الكتروني يحوي بعض تلك العناصر.

الفكرة العامة تتلخص في ربط نظام الأندرويد مع المتحكم أردوينو واستقبال الاوامر المرسلة من نظام الأندرويد والتفاعل معها من قبل المتحكم لتطبيق الأوامر اللازمة على العناصر المرتبطة مع هذا المتحكم ايضا قراءة قيم الحساسات / التراسونيك / ومعالجتها و من ثم ارسال قيم هذه القراءات بشكل معاكس الى نظام الاندرويد الذي سوف يقوم بإظهار هذه القيم للمستخدم كما سوف نرى في الفقرات القادمة.

قمنا ببلورة الأفكار السابقة من خلال بناء روبوت بسيط هو عبارة عن سيارة تحوي محركين مستمرين ودارة قيادة لتلك المحركات المحددة بمواصفات خاصة ك جهد الدخل أو التيار المستجر من تلك الشريحة وبعض الليدات وبإضافة حساس مسافة لقياس اقرب حاجز عن الروبوت اضافة الى موديول البلوتوث المسؤول عن تحقيق عملية الاتصال وتامين تراسل البيانات ومكبر صوت لإظهار التنبيهات وذراع سيرفو يتم التحكم بها عن طريق دمج الكود التطبيقي الذي نفذناه لها ضمن كود تطبيق الأردينو ليغدو التطبيق تطبيق أندرويد واحد لكليهما .

أما في نظام الأندرويد حققنا عملية التحكم بأكثر من اسلوب عبر برمجة تطبيق متعدد الواجهات هذا التطبيق يتفاعل مع المستخدم بأحدث واسهل التقنيات الموجودة حاليا مثل تنفيذ الاوامر الصوتية التي يتيحها النظام عبر خاصية Google للتعرف على الصوت واستخدام الخرج بناءً عليه لتحديد القرار (كلمة صوتية = أمر تحكم معين) أو استخدام حساسات التسارع جايروسكوب Accelerometer ، التي تقيس قيم مقدار تغير ميول الجهاز على المحاور الاحداثية الثلاثة x , y , z أو استخدام الطريقة التقليدية المتمثلة بواجهة تحوي مجموعة من الازرار بعدها سوف يقوم هذا التطبيق بالوصول الى الموارد الصلبة للجهاز الذكي واستخدام البلوتوث.

ولقد قمنا ببناء تطبيق آخر خاص فقط بشريحة IOIO أسميناه RobotControl عن طريقه يتم التحكم بـJoyStickClass بسرعة واتجاه دوران أربع محركات

مستمرة بمبدأ التعديل العرضي المطالي عن طريق دارة lm298 ونظراً لعدم توفر قطع كافية ولبعض المشاكل التقنية والمادية ولكوننا فقط سنعرض تطبيقاً عملياً واحداً وفق آلية توصيل معينة ففضلنا التطبيق الأقوى المشروح سابقاً بالبداية ولكن بما أنَّ كود الـ RobotControl كان أيضاً ناجح فقط يحتاج لدارة أخرى عملية وكلف مادية جديدة ووقت أكثر للتطوير والتحسين ولا يسعنا الآن إلا أن نحقق على أرض الواقع تطبيقاً واحداً فسوف ألحق كود الـ RobotControl بملفات الأطروحة لتعم الفائدة خاصة لمن أحب مجال الجافا وبرمجة اليويو بورد.

الفصل الرابع

١.١ القسم العملي التنفيذي للمشروع :

١ قسم الأكواد CODES :

١.١ الكود الخاص بالمتحكم Arduino:

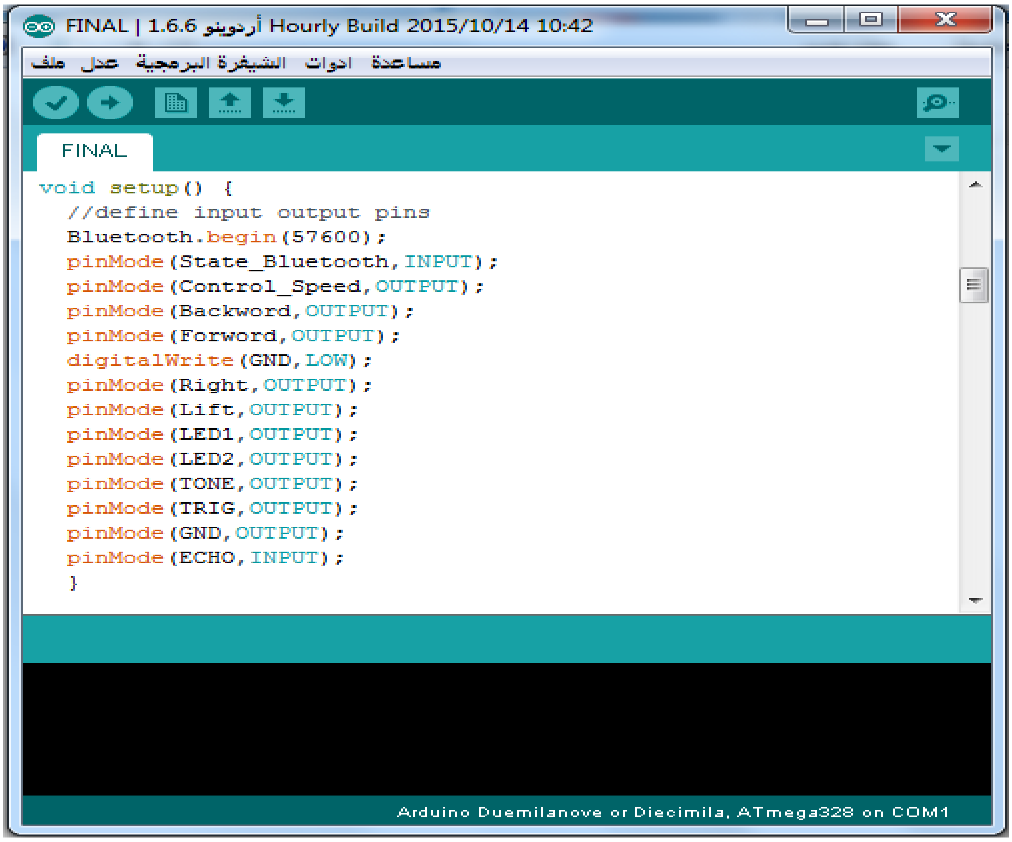
FINAL | 1.6.6 أردوينو Hourly Build 2015/10/14 10:42

ملف عدل الشيفرة البرمجية ادوات مساعدة

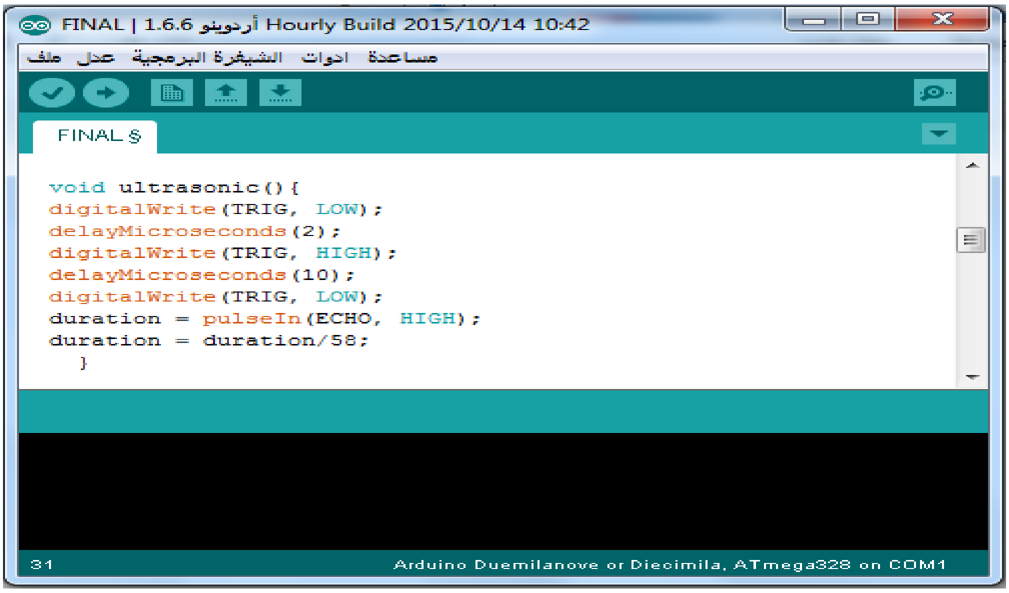
FINAL

```
// import softwareserial library
#include <SoftwareSerial.h>
// difine variables
SoftwareSerial Bluetooth(0,1);
unsigned long mflisher = millis();
unsigned long mtone = millis();
unsigned long mUltra = millis();
boolean isRun=false;
int State_Bluetooth=2;
int Control_Speed=3;
int BluetoothData;
int Forword=13;
int Backword=12;
int Right=11;
int Lift=10;
int TRIG=6;
int ECHO=7;
int LED1=4;
int LED2=5;
int TONE=9;
int GND=8;
int bFrontLed = 0;
int bFlisher = 0;
int bbFlisher = 0;
int stateBluetooth = 0;
int readUltea = 0;
int btone = 1;
long duration;
char buf[4];
```

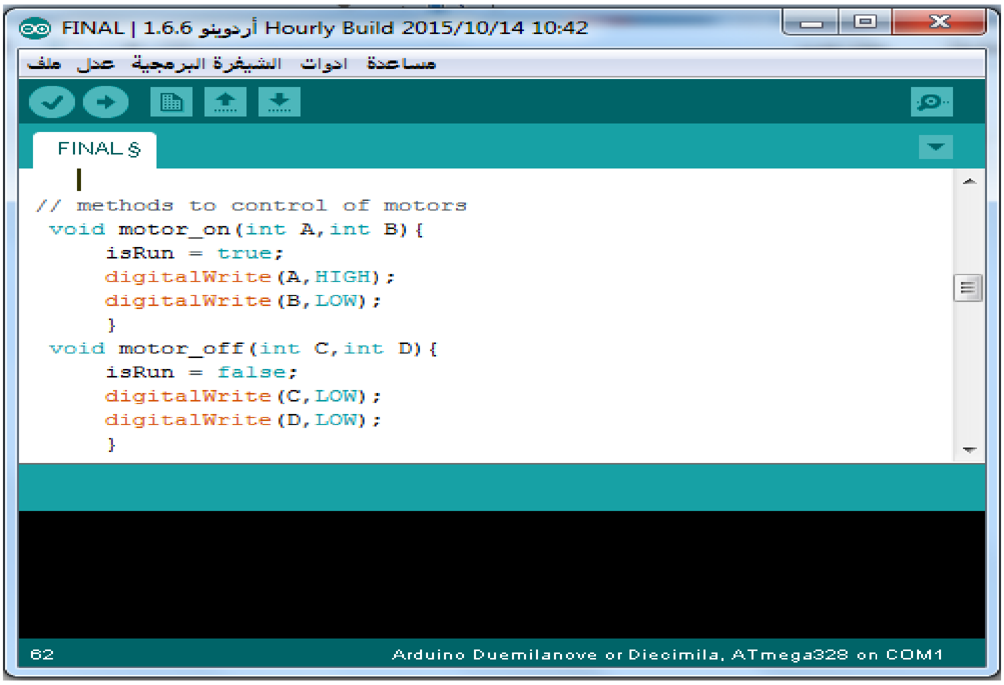
الشكل (٣ - ١) تهيئة المتغيرات اللازمة ضمن الكود



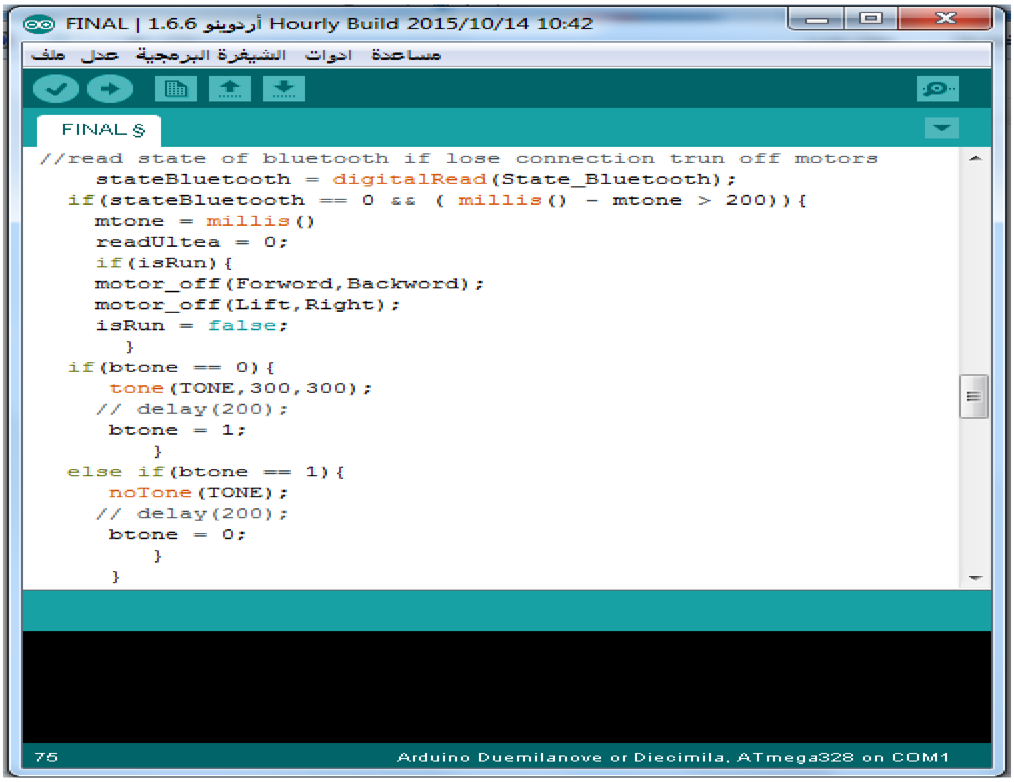
الشكل (٣ - ٢) تهيئة منافذ الدخل و الخرج للمتحكم



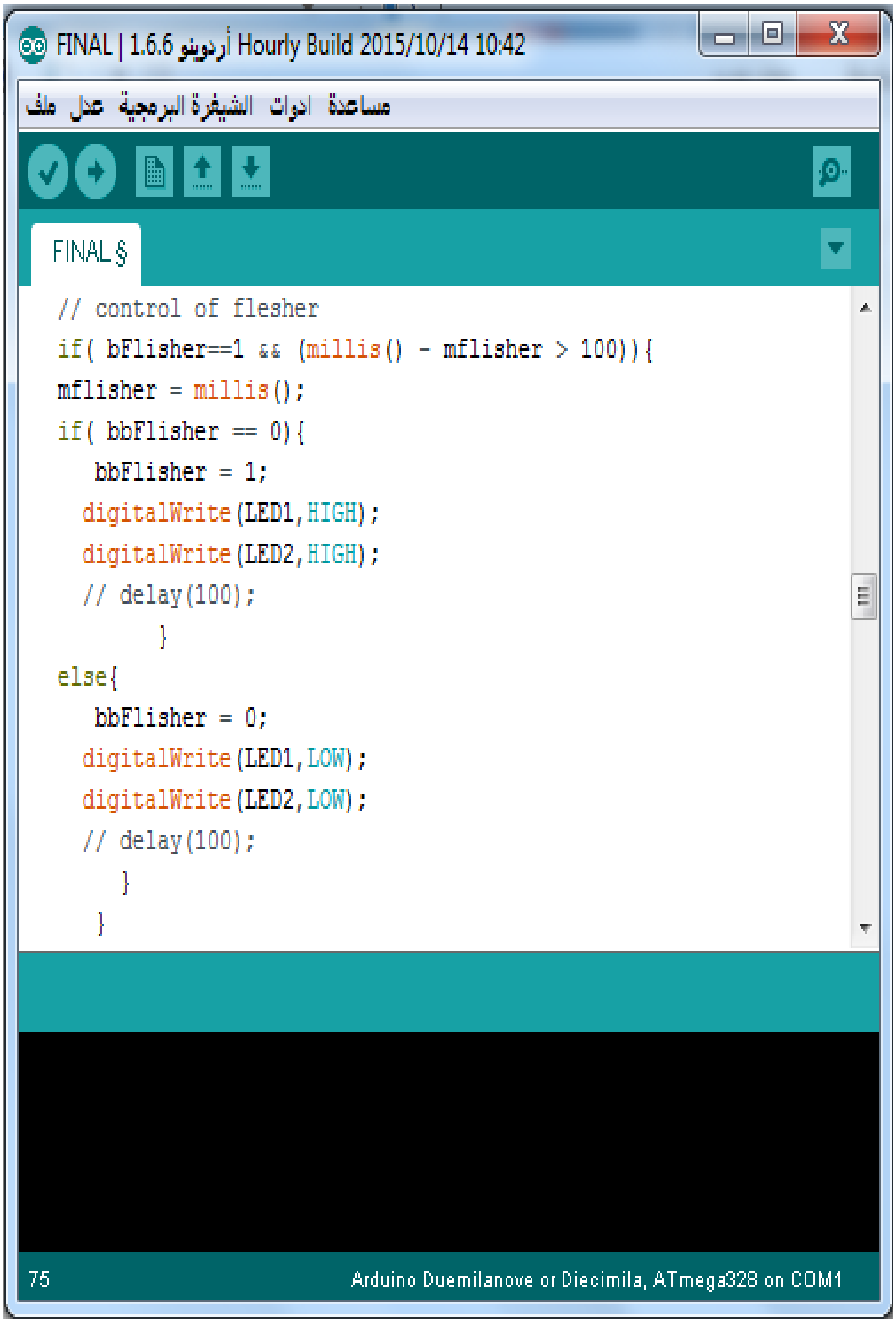
الشكل (٣ - ٣) قراءة قيم حساس المسافة



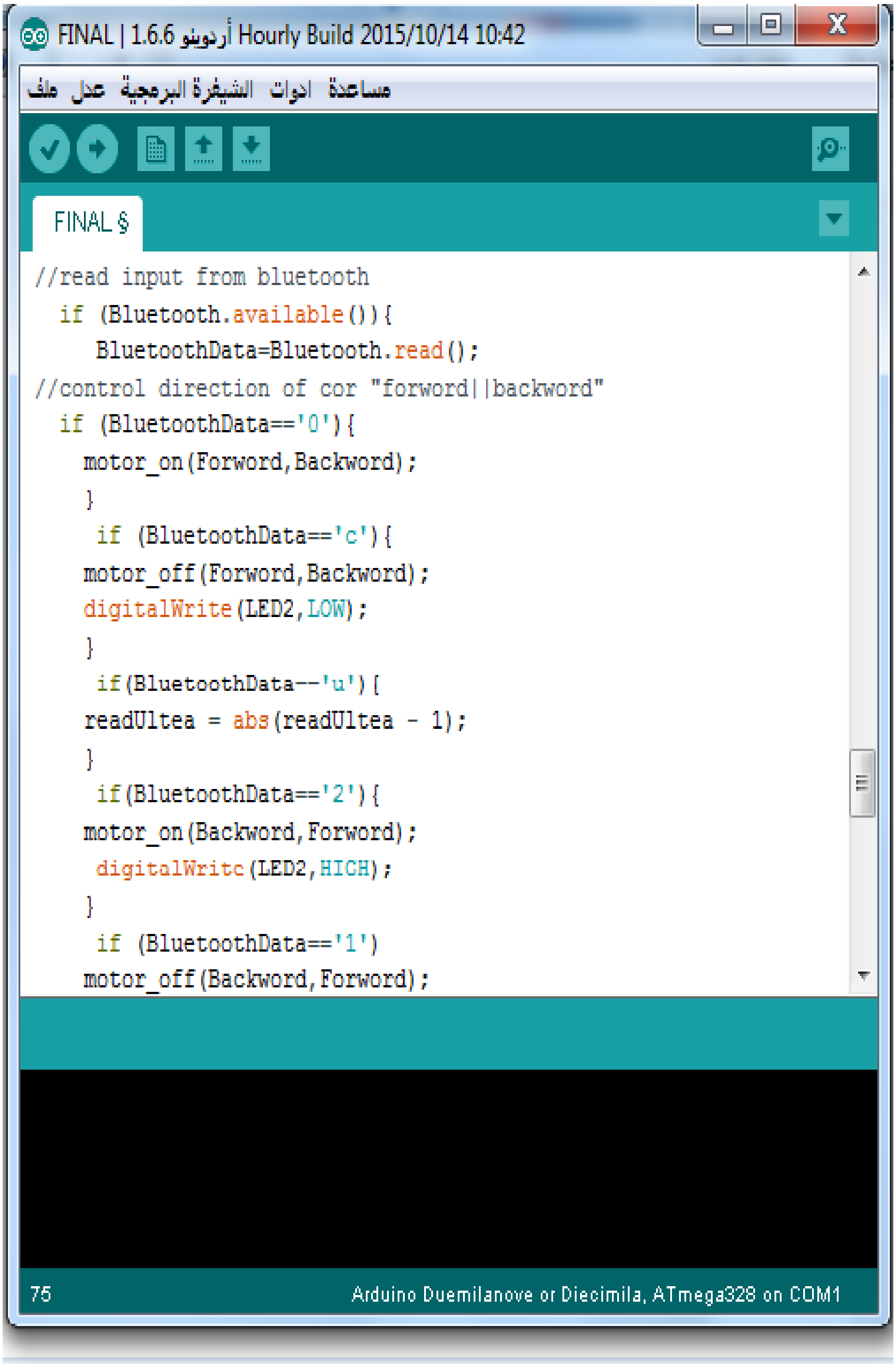
الشكل (٣ - ٤) الجزء الخاص بتشغيل و اطفاء المحركات



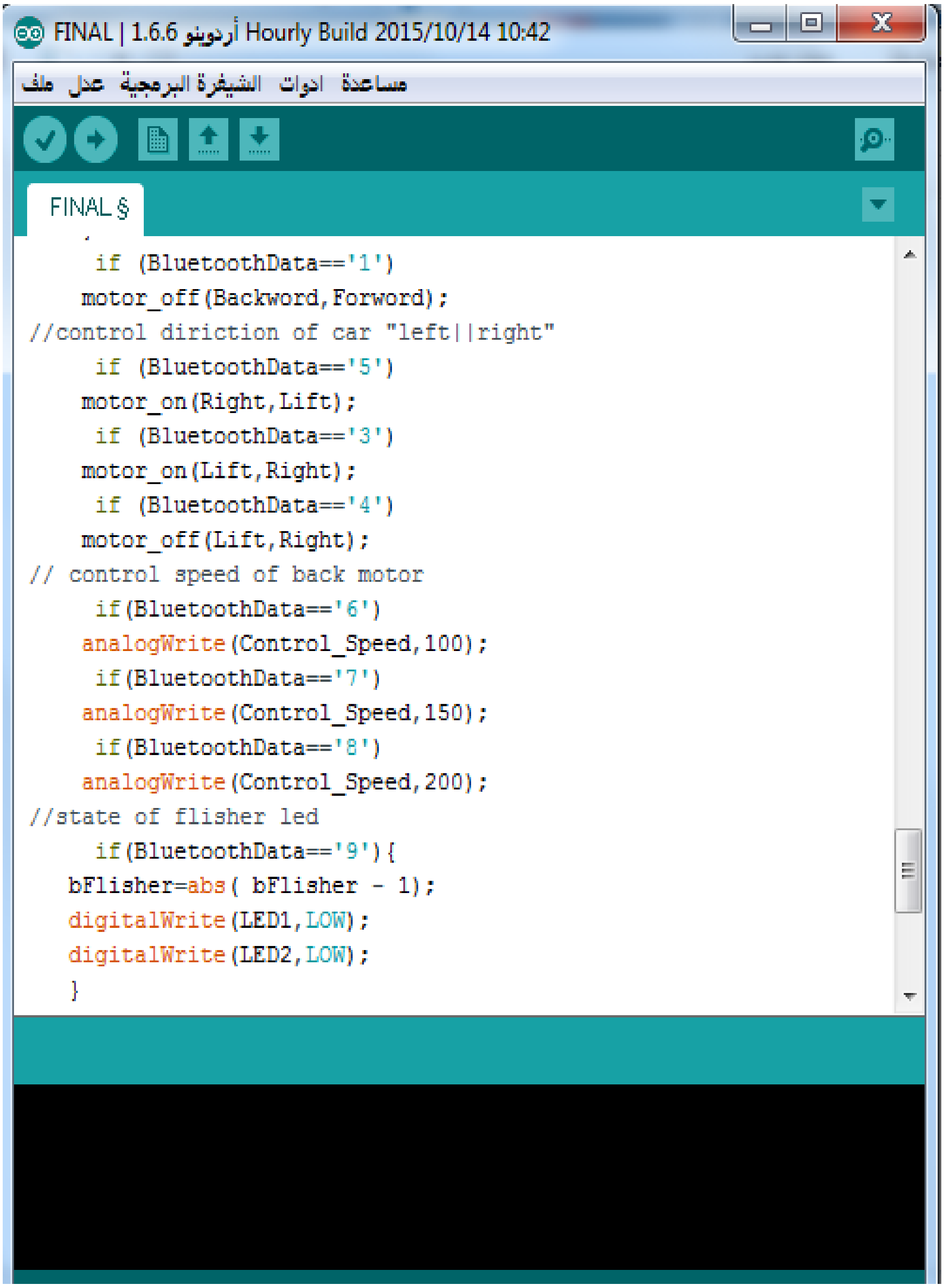
الشكل (٣ - ٥) قراءة قيمة الحالة لشريحة البلوتوث لمعرفة وضعية الاتصال وتشغيل المنبه بحالة فقدان الاتصال



الشكل (٣ ـ ٦) الجزء الخاص بتشغيل واطفاء الليدات بحال تفعيل ذلك الأمر

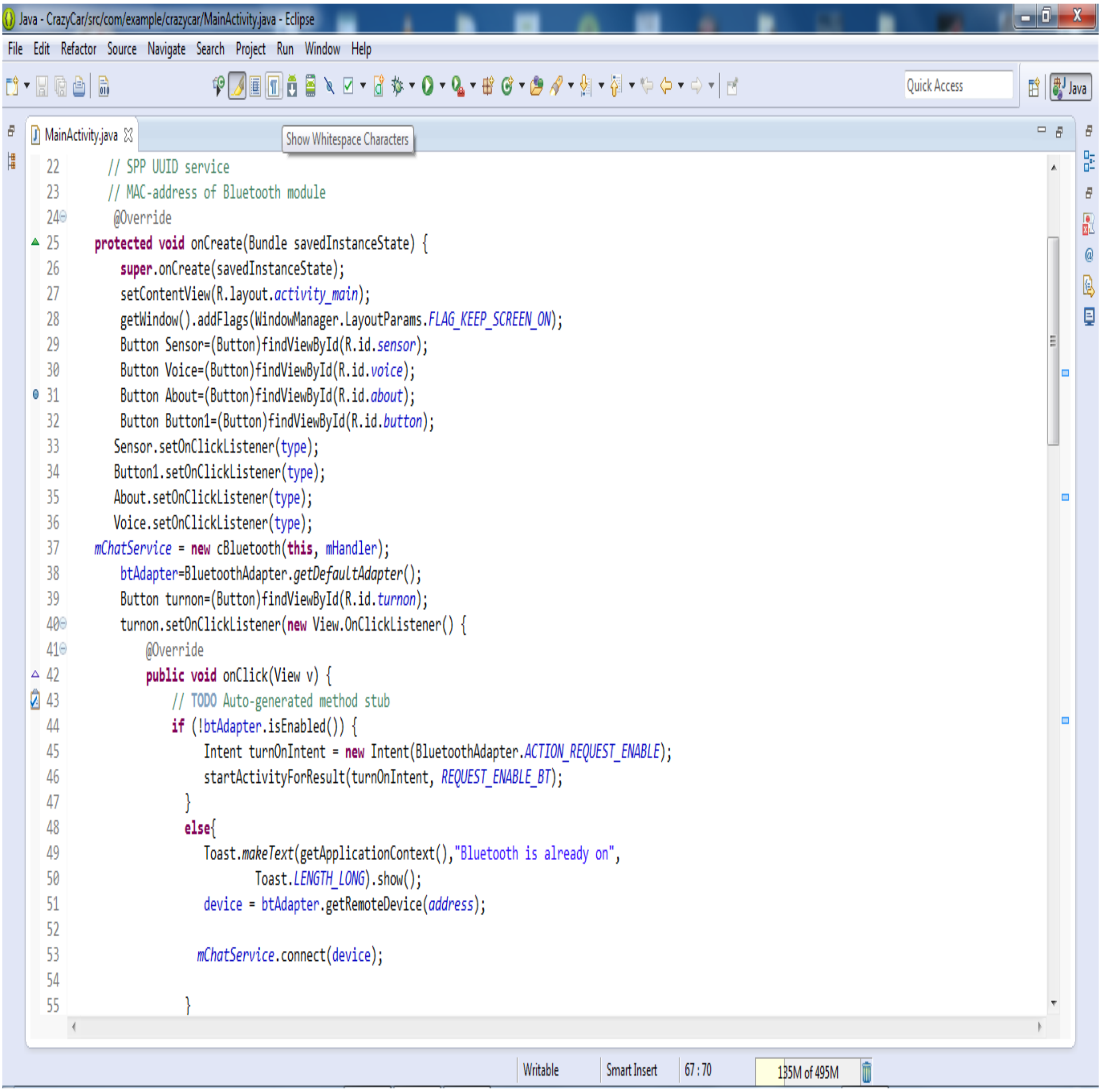


الشكل (٣ - ٧)الجزء الخاص بقراءة الرمز المستقبل من الهاتف في حال كان الاتصال متاحا ومن ثم تنفيذ الأمر المطلوب



الشكل (٣ ـ ٨) تتمة الجزء الخاص بقراءة الرمز المستقبل من الهاتف في حال كان الاتصال متاحا ومن ثم تنفيذ الأمر المطلوب

١.٢ الكود الخاص بنظام أندرويد

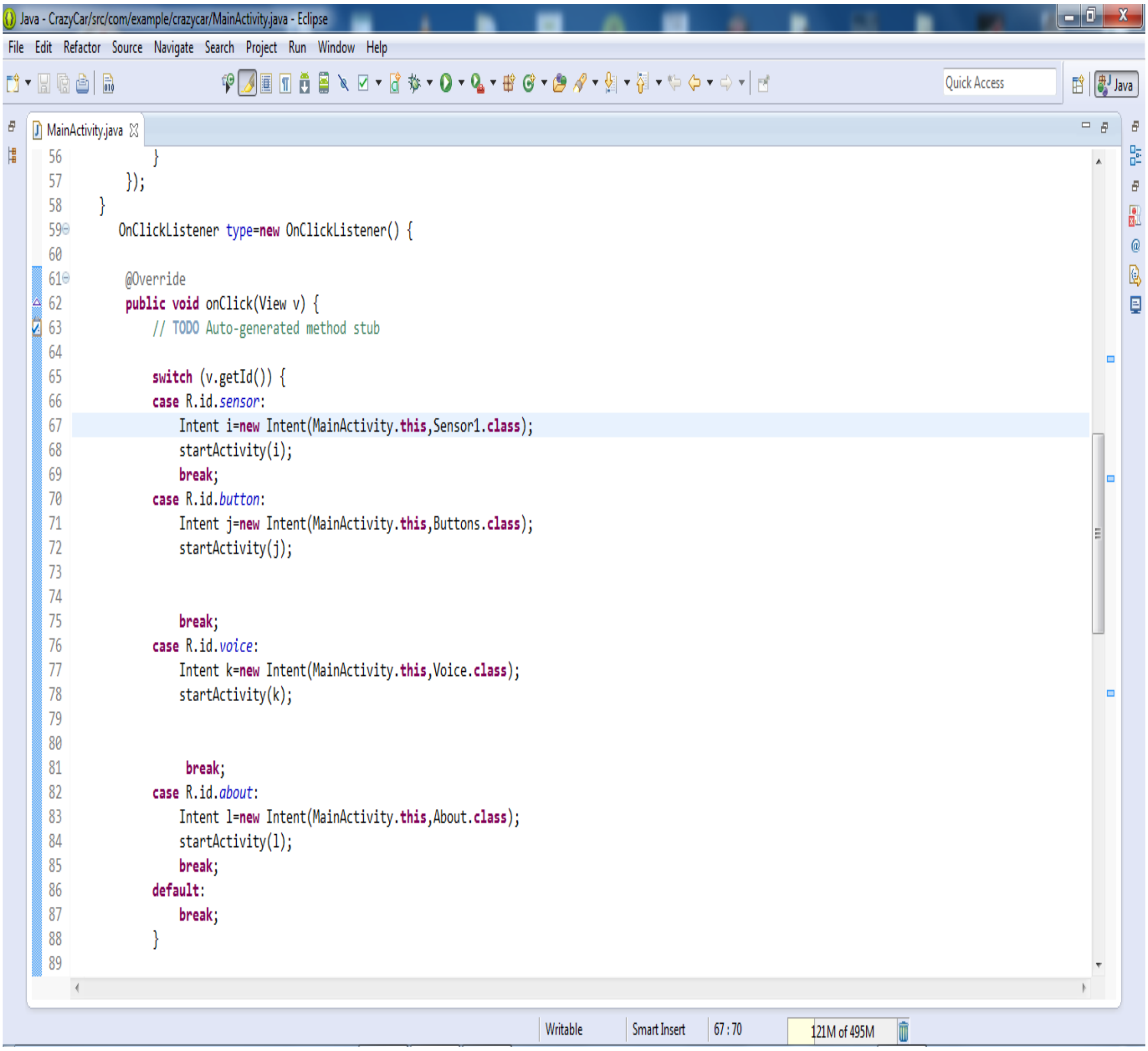



```java
    // SPP UUID service
    // MAC-address of Bluetooth module
     @Override
  protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
      getWindow().addFlags(WindowManager.LayoutParams.FLAG_KEEP_SCREEN_ON);
      Button Sensor=(Button)findViewById(R.id.sensor);
      Button Voice=(Button)findViewById(R.id.voice);
      Button About=(Button)findViewById(R.id.about);
      Button Button1=(Button)findViewById(R.id.button);
     Sensor.setOnClickListener(type);
     Button1.setOnClickListener(type);
     About.setOnClickListener(type);
     Voice.setOnClickListener(type);
  mChatService = new cBluetooth(this, mHandler);
      btAdapter=BluetoothAdapter.getDefaultAdapter();
      Button turnon=(Button)findViewById(R.id.turnon);
      turnon.setOnClickListener(new View.OnClickListener() {
          @Override
          public void onClick(View v) {
              // TODO Auto-generated method stub
              if (!btAdapter.isEnabled()) {
                  Intent turnOnIntent = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
                  startActivityForResult(turnOnIntent, REQUEST_ENABLE_BT);
              }
              else{
                  Toast.makeText(getApplicationContext(),"Bluetooth is already on",
                          Toast.LENGTH_LONG).show();
                  device = btAdapter.getRemoteDevice(address);

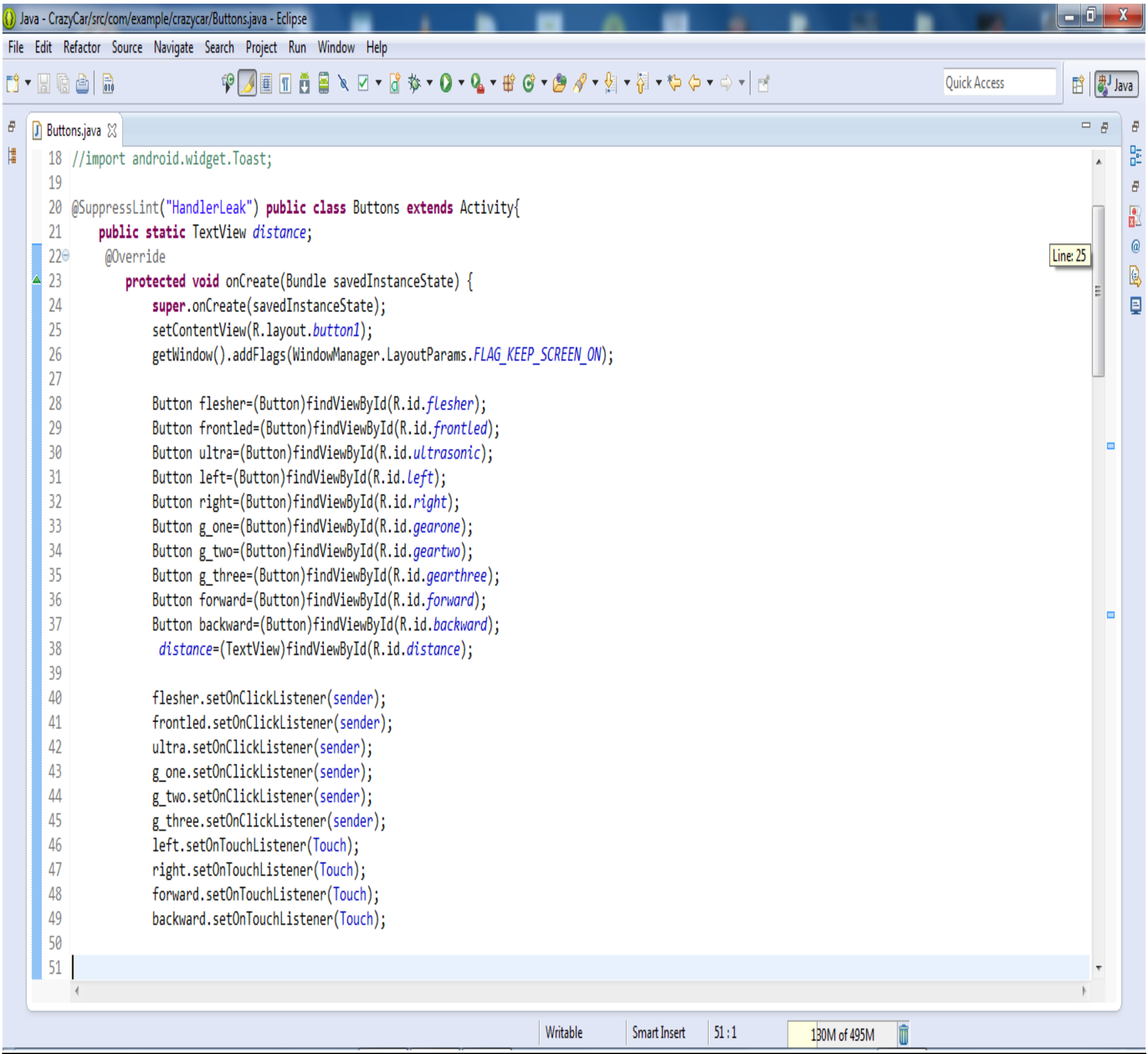
                 mChatService.connect(device);

              }
```

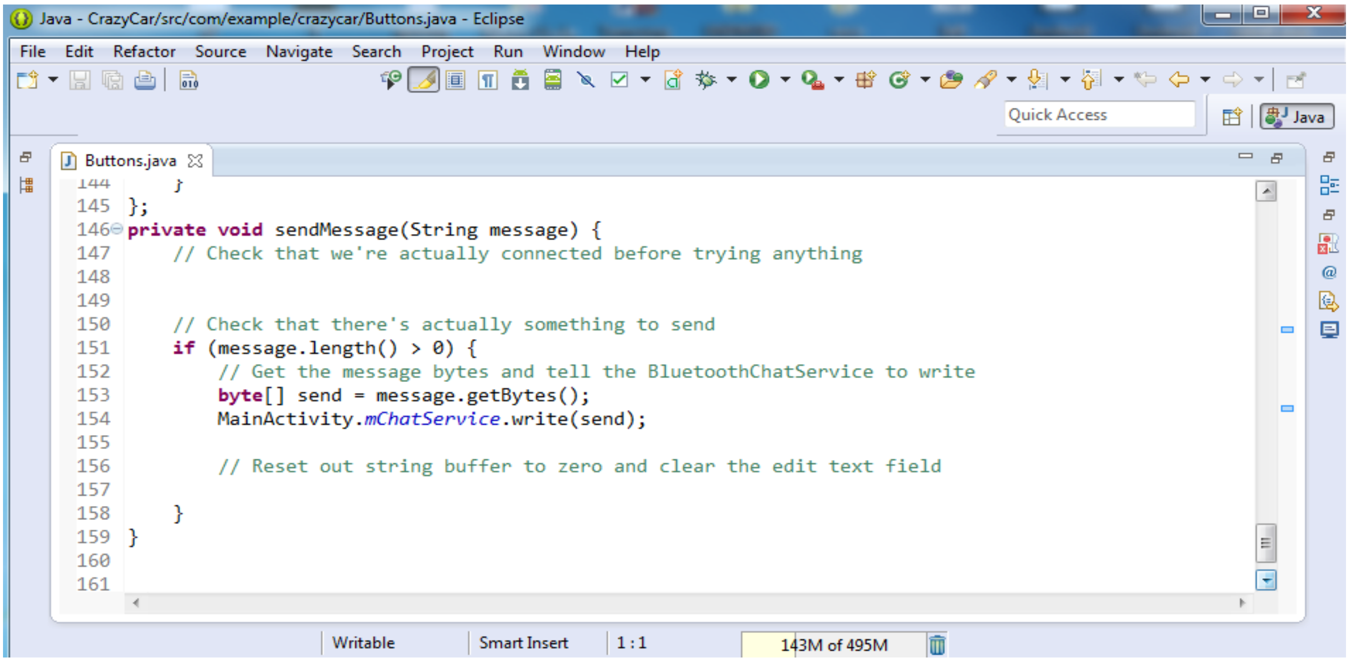
الشكل (٤ - ١) جزء الكود الخاص بتهيئة عناصر الواجهة الرئيسية وتشغيل البلوتوث وتمكين الاتصال



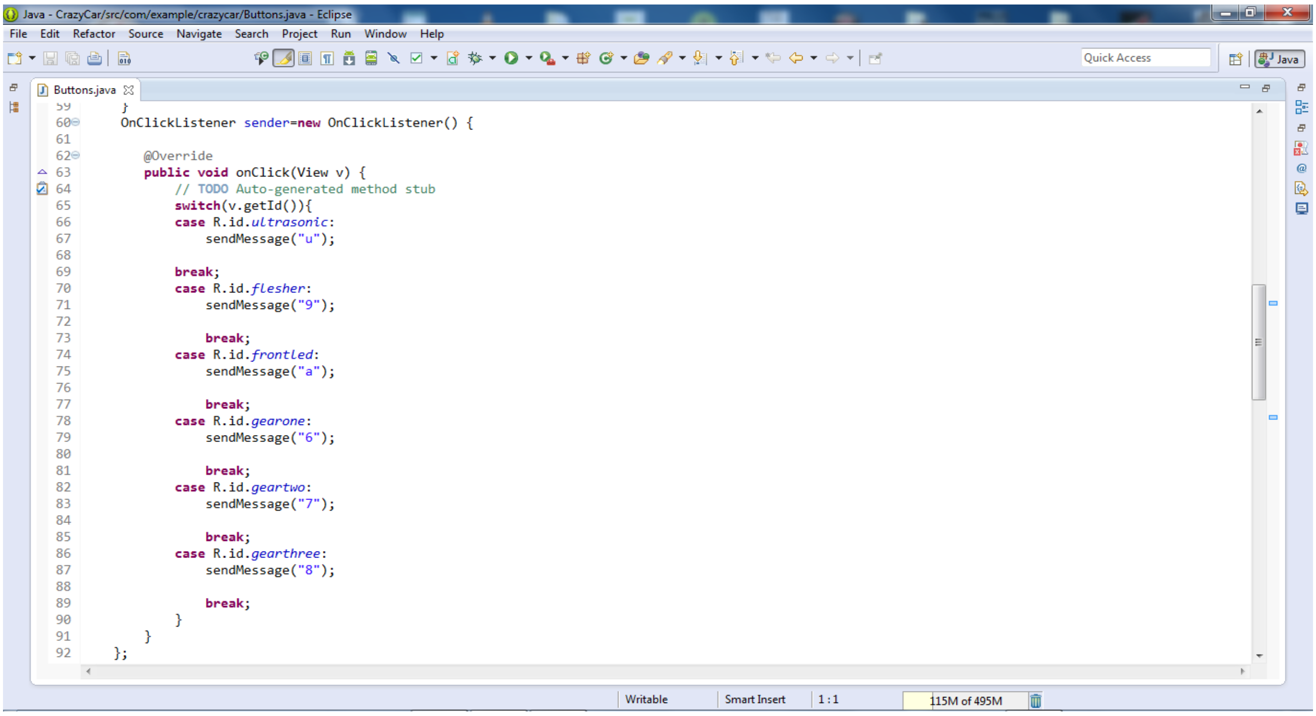
الشكل (٤ - ٢) جزء الكود الخاص باختيار نمط التحكم



الشكل(٤ ـ ٣)جزء الكود الخاص بنمط تحكم الأزرار مع التهيئات

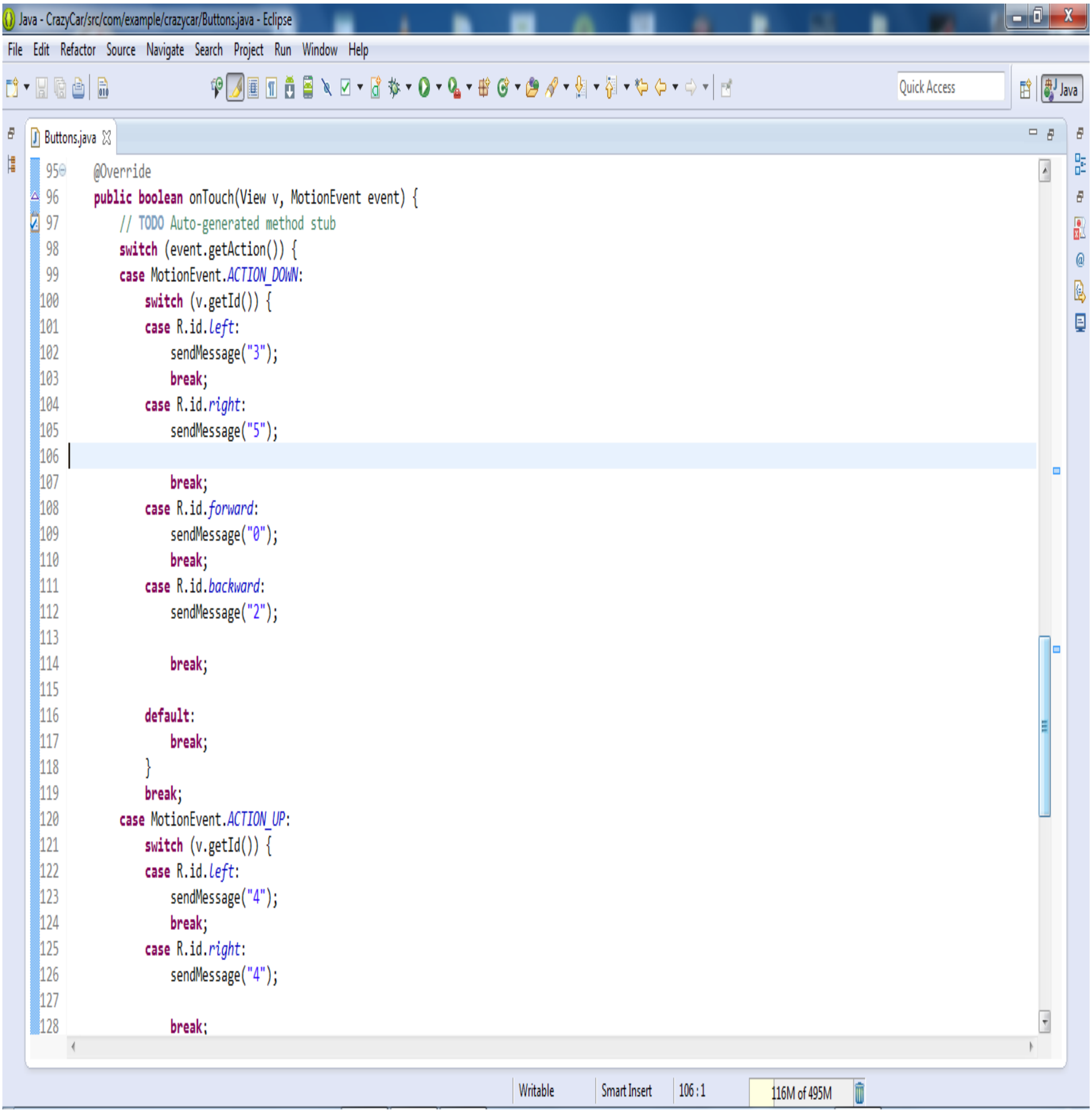


الشكل (٤ ـ ٤)التابع الذي يقوم بإرسال الأوامر الى المتحكم



الشكل (٤ – ٥)الجزء المسؤول عن تحديد الزر المضغوط ثم إرسال الأمر المناسب

حدث *OnClick*



```java
@Override
public boolean onTouch(View v, MotionEvent event) {
    // TODO Auto-generated method stub
    switch (event.getAction()) {
    case MotionEvent.ACTION_DOWN:
        switch (v.getId()) {
        case R.id.left:
            sendMessage("3");
            break;
        case R.id.right:
            sendMessage("5");

            break;
        case R.id.forward:
            sendMessage("0");
            break;
        case R.id.backward:
            sendMessage("2");

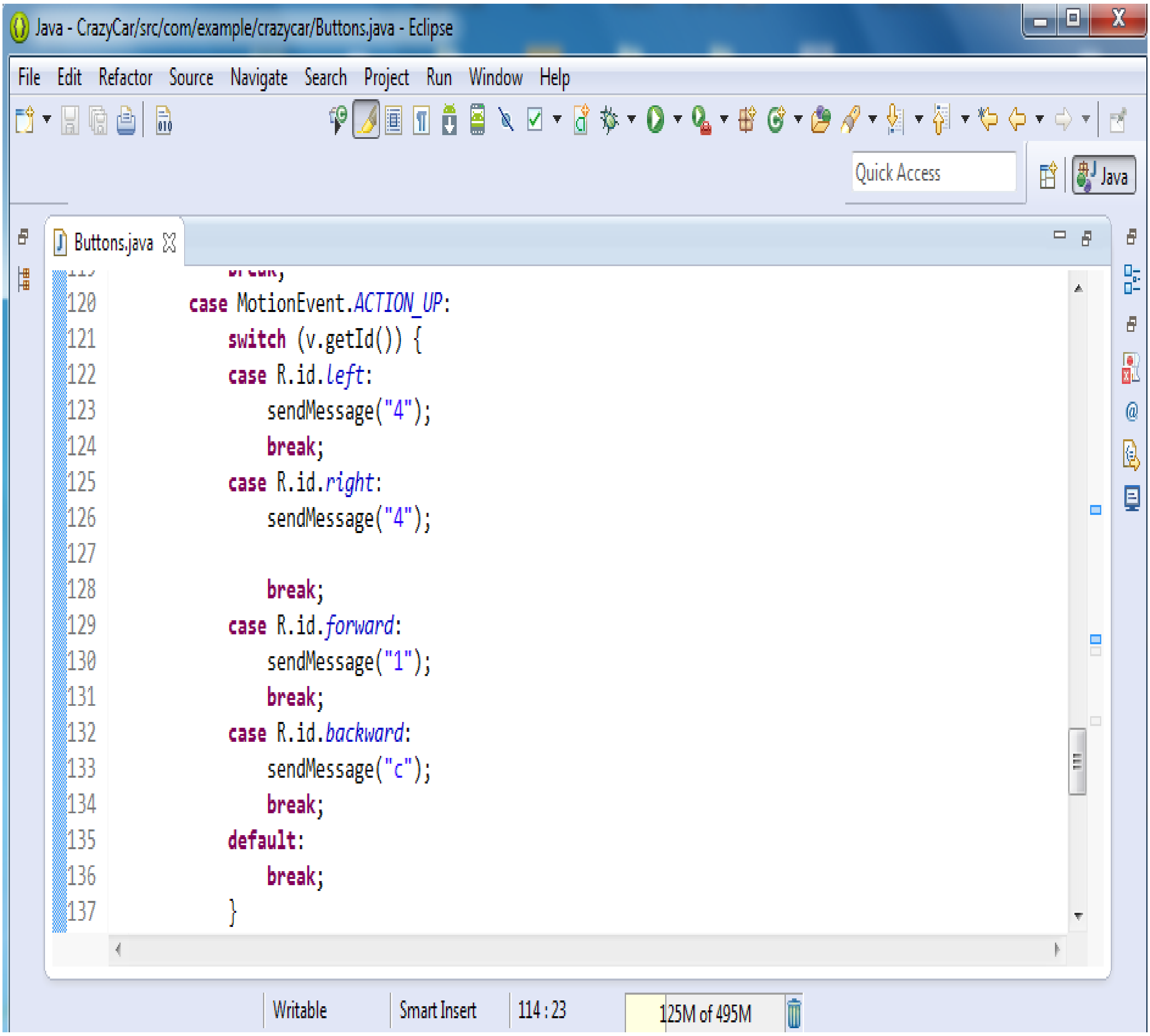
            break;

        default:
            break;
        }
        break;
    case MotionEvent.ACTION_UP:
        switch (v.getId()) {
        case R.id.left:
            sendMessage("4");
            break;
        case R.id.right:
            sendMessage("4");

            break;
```

الشكل (٤ ـ ٦)الجزء المسؤول عن تحديد الزر المضغوط ثم إرسال الأمر المناسب

حدث *OnTouch & Action Down*



الشكل (٤ ـ ٧) الجزء المسؤول عن تحديد الزر المضغوط ثم إرسال الأمر المناسب

حدث *OnTouch* **&** *Action Up*

yasserservo.java

```
// Servos Code with Gyro
//YasserKarKouKli & Muayaad Shiekh Amin

    import ioio.lib.api.PwmOutput;
import ioio.lib.api.exception.ConnectionLostException;
import ioio.lib.util.BaseIOIOLooper;
import ioio.lib.util.IOIOLooper;
import ioio.lib.util.android.IOIOActivity;

import android.R;
import android.content.Context;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;
import android.os.Bundle;
import android.util.Log;
import android.widget.SeekBar;
import android.widget.TextView;
import android.widget.ToggleButton;
import android.widget.SeekBar.OnSeekBarChangeListener;
    public class yasserservo extends IOIOActivity implements SensorEventListener
    {
        static final String TAG = "yasser_RC";

        /*****************************************************************  GUI
*****************************************************************/
        ToggleButton toggleDrive;
        TextView accelView = null;
        TextView compassView = null;
        TextView gyroView = null;
        private SeekBar seekBar_;

        /*****************************************************************  sensors
*****************************************************************/
        SensorManager mSensorManager;
        Sensor accel;
        Sensor compass;
        Sensor gyro;

        /*****************************************************************  Values for
orientation and acceleration
*****************************************************************/
        float[] mMagneticValues;
        float[] mAccelerometerValues;
        float[] gyroscope_values;
        float[] orientation;
        float[] Rot;

        /*****************************************************************  IOIO
*****************************************************************/
        static final int MOTOR_PIN = 5;
        static final int SERVO_PIN = 35;

        /*****************************************************************  Values for
controlling car   *****************************************************************/
```

Page 1

yasserservo.java

```
	/******************************************************************* activity
*******************************************************************/

	@Override
	protected void onCreate(Bundle savedInstanceState)
	{
	    super.onCreate(savedInstanceState);
	    setContentView(R.layout.activity_list_item);

	    mSensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
	    accel = mSensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
	    compass = mSensorManager.getDefaultSensor(Sensor.TYPE_MAGNETIC_FIELD);
	    gyro = mSensorManager.getDefaultSensor(Sensor.TYPE_GYROSCOPE);

	    accelView = (TextView) findViewById(R.id.textAccel);
	    compassView = (TextView) findViewById(R.id.textCompass);
	    gyroView = (TextView) findViewById(R.id.textGyro);
	    toggleDrive = (ToggleButton) findViewById(R.id.drive_btn);
	    seekBar_ = (SeekBar)findViewById(R.id.seekBar1);
	}

	@Override
	protected void onResume()
	{
	    super.onResume();
	    mSensorManager.registerListener(this,
accel,SensorManager.SENSOR_DELAY_GAME);
	    mSensorManager.registerListener(this,
compass,SensorManager.SENSOR_DELAY_GAME);
	    mSensorManager.registerListener(this,
gyro,    SensorManager.SENSOR_DELAY_GAME);

	    Rot = new float[9];
	    mMagneticValues = new float[3];
	    mAccelerometerValues = new float[3];
	    orientation = new float[3];
	    gyroscope_values = new float[3];
	}

	@Override
	protected void onPause()
	{
	    super.onPause();
	    mSensorManager.unregisterListener(this);
	}


	/******************************************************************* sensors
*******************************************************************/
	@Override
	public void onSensorChanged(SensorEvent event)
	{
	    switch (event.sensor.getType())
	    {
	    case Sensor.TYPE_MAGNETIC_FIELD:
	        mMagneticValues[0] = event.values[0];
```

Page 2

yasserservo.java

```
                gyroscope_values[0] = event.values[0];
                gyroscope_values[1] = event.values[1];
                gyroscope_values[2] = event.values[2];
                gyroView.setText("Gyro: " + "\n"+ gyroscope_values[0] + "\n" +
gyroscope_values[1] +"\n"+ gyroscope_values[2]);
                break;

            case Sensor.TYPE_ACCELEROMETER:
                mAccelerometerValues[0] = event.values[0];
                mAccelerometerValues[1] = event.values[1];
                mAccelerometerValues[2] = event.values[2];
                accelView.setText("Accel: " + "\n"+ mAccelerometerValues[0] + "\n" +
mAccelerometerValues[1] +"\n"+ mAccelerometerValues[2]);

                p_servo = (int) (mAccelerometerValues[0] * COEFF_SERVO);
                p_motor = (int) (mAccelerometerValues[1] * COEFF_MOTOR);
                break;
            }

            SensorManager.getRotationMatrix(Rot, null, mAccelerometerValues,
mMagneticValues);
            SensorManager.getOrientation(Rot, orientation);
            orientation[0] = (float) Math.toDegrees(orientation[0]);
            orientation[1] = (float) Math.toDegrees(orientation[1]);
            orientation[2] = (float) Math.toDegrees(orientation[2]);
            compassView.setText("Compass: " + "\n"+ orientation[0] + "\n" +
orientation[1] +"\n"+ orientation[2]);
        }

        @Override
        public final void onAccuracyChanged(Sensor sensor, int accuracy) {}


        /****************************************************************  IOIO
****************************************************************/
        class Looper extends BaseIOIOLooper
        {
            PwmOutput motor;    // motor to move forward and backward
            PwmOutput servo,pwm35;  // servo to steer/turn

            @Override
            protected void setup() throws ConnectionLostException
            {
                motor = ioio_.openPwmOutput(MOTOR_PIN, 100);
                servo = ioio_.openPwmOutput(SERVO_PIN, 100);


                pwm35.setPulseWidth(3100 - 650 - seekBar_.getProgress());
                statLed = ioio_.openPwmOutput(0, 100);}

            @Override
            public void loop() throws ConnectionLostException
            {
                try
                {
                     seekBar_.setOnSeekBarChangeListener(new OnSeekBarChangeListener()
```

Page 3

yasserservo.java

```
                    @Override
                    public void onStartTrackingTouch(SeekBar seekBar) {
                    }

                    @Override
                    public void onProgressChanged(SeekBar seekBar, int
progress,boolean fromUser) {
                        textView_.setText("PWM Frequenz: " +
String.valueOf(3100 - 650 - seekBar_.getProgress()));
                        pwm35.setPulseWidth(3100 - 650 -
seekBar_.getProgress());
                        PwmOutput statLed;
                        statLed.setPulseWidth(seekBar_.getProgress()*6); }
                });
                if (toggleDrive.isChecked())                // transition from off
to on
                {
                    pwm_motor = PWM_STOP - p_motor;
                    pwm_servo = PWM_STOP - p_servo;

                    if(REVERSE_PWM == true)
                    {
                        pwm_motor = PWM_STOP - (pwm_motor-PWM_STOP);
                        pwm_servo = PWM_STOP - (pwm_servo-PWM_STOP);
                    }
                }
                else            // transition from on to off
                {
                    pwm_motor = PWM_STOP;
                    pwm_servo = PWM_STOP;
                }
//              Log.d(TAG, "PWM: " + pwm_motor + "  " + pwm_servo);

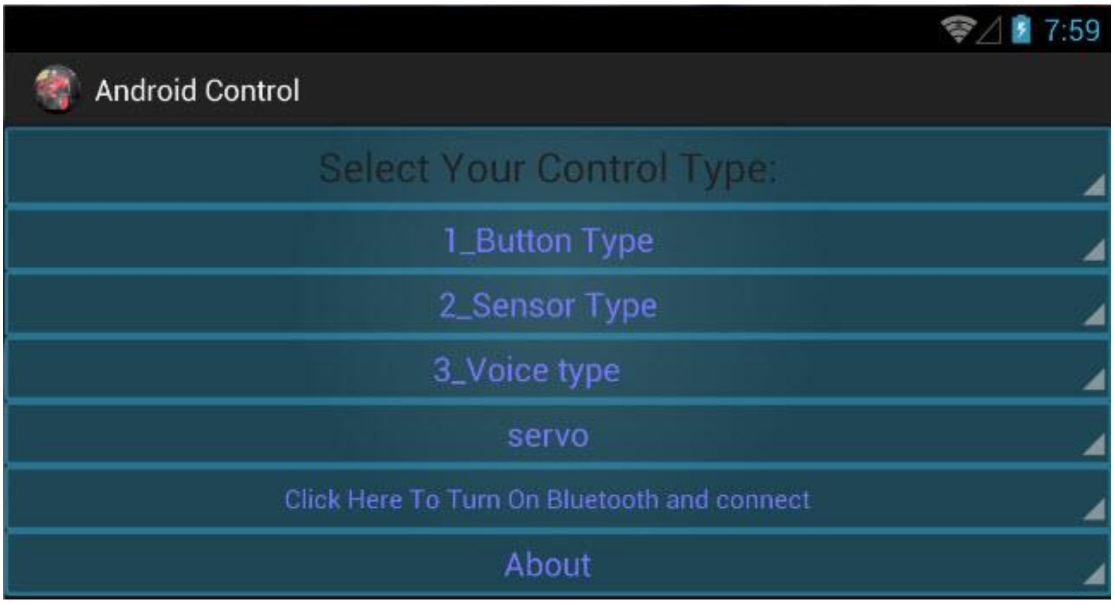
                motor.setPulseWidth(pwm_motor);
                servo.setPulseWidth(pwm_servo);
                Thread.sleep(10);
            }
            catch (InterruptedException exc) {Log.e(TAG, "Error: ", exc);   }
        }
    }

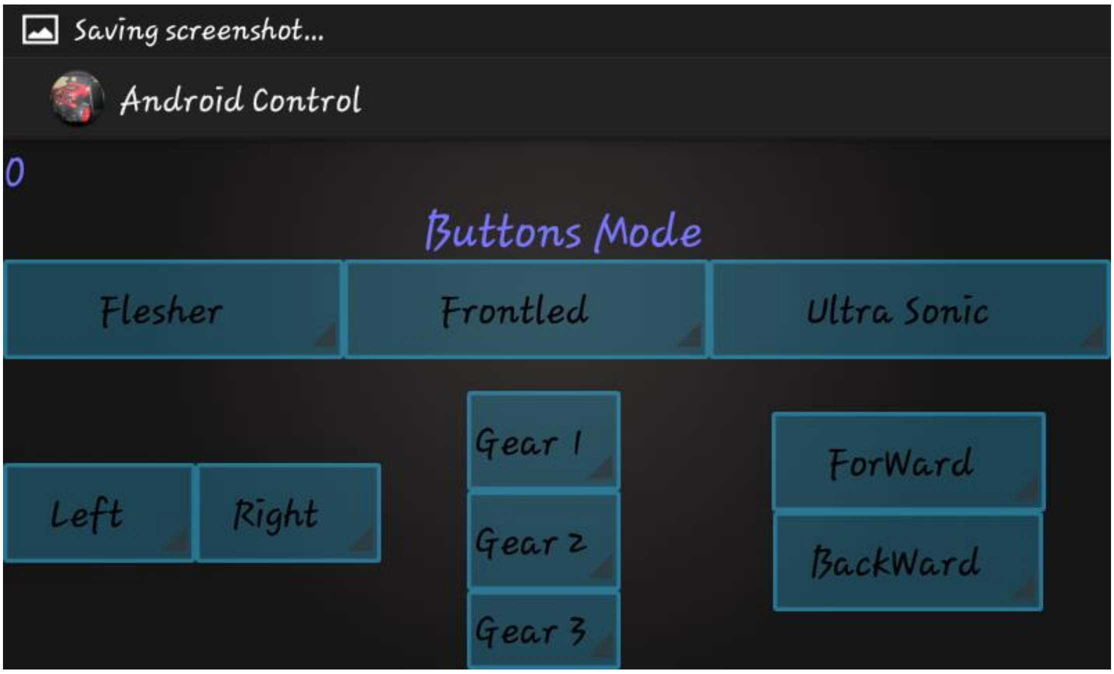
    @Override
    protected IOIOLooper createIOIOLooper() {
        return new Looper();
    }
}
```

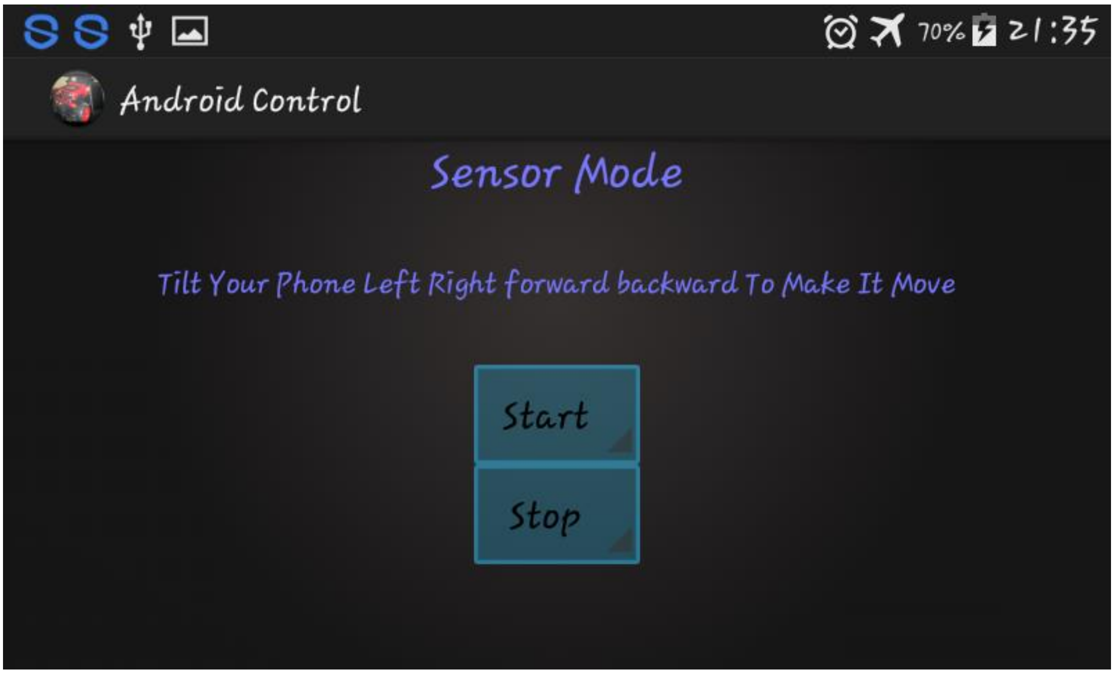
Page 4

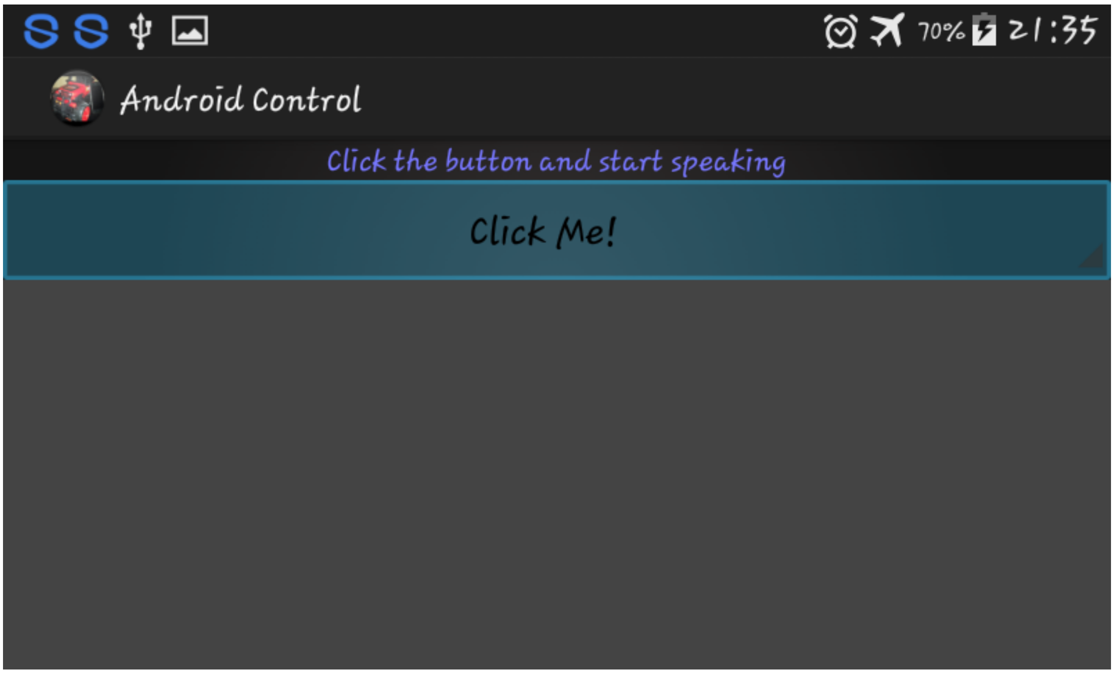
الشكل (٤ -٨) جزء الكود الخاص بتهيئة عناصر الواجهة الرئيسية وتشغيل البلوتوث وتمكين الاتصال للتحكم بذراع السيرفو

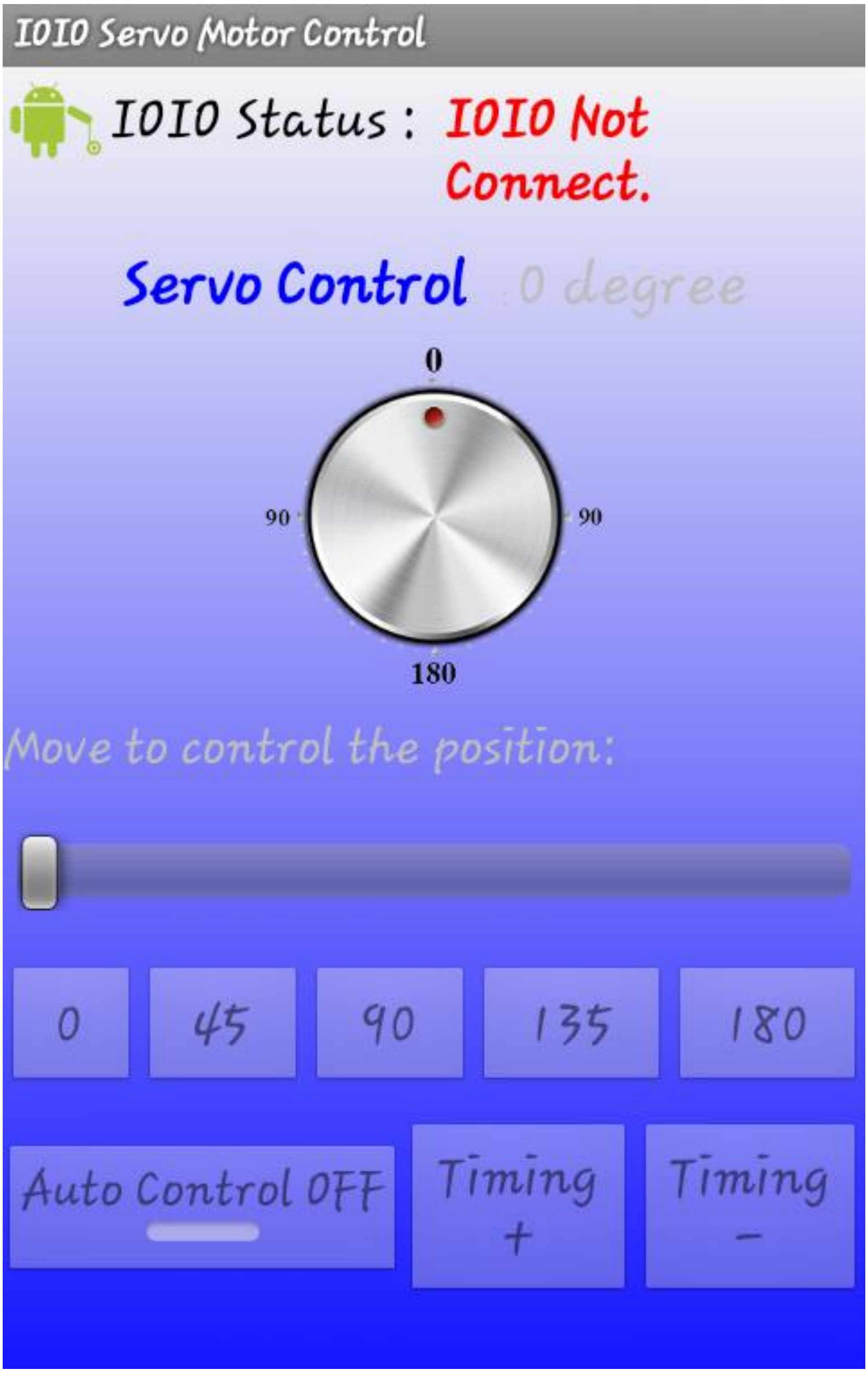
٢ واجهات تطبيق الأندرويد:



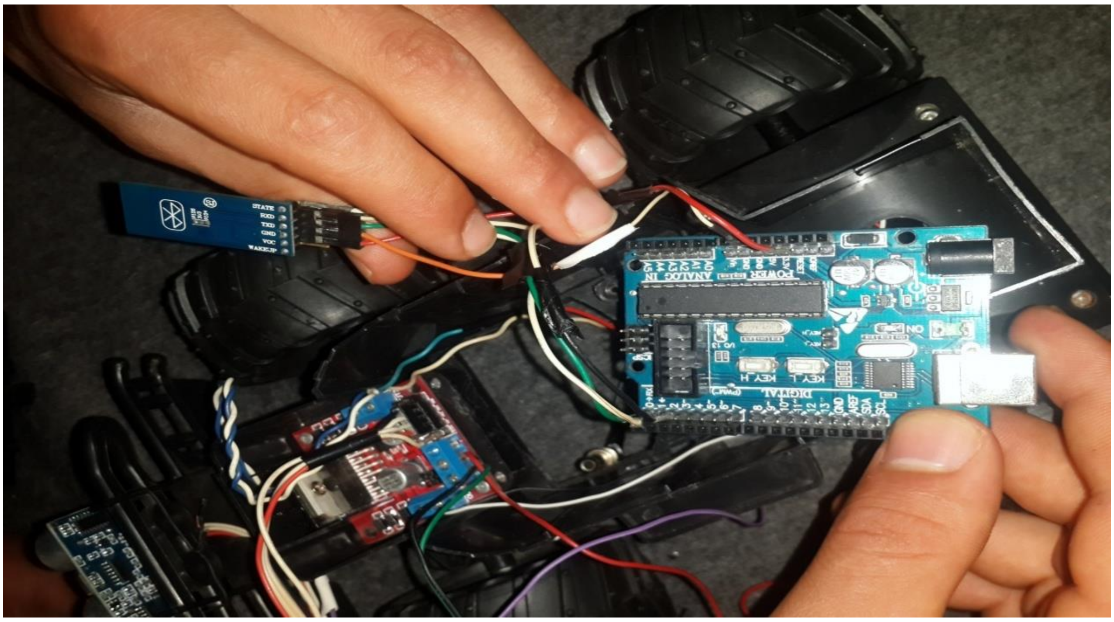


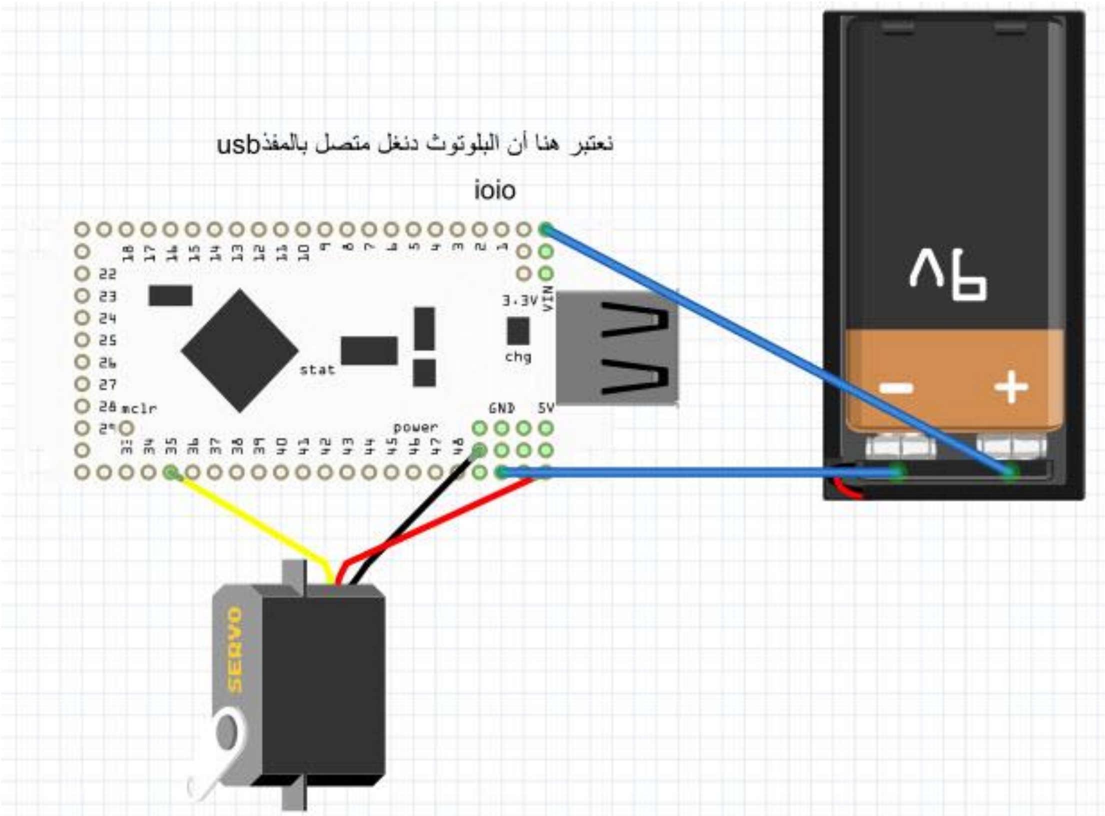

Android Control
Sensor Mode
Tilt Your Phone Left Right forward backward To Make It Move
Start
Stop


Android Control
Click the button and start speaking
Click Me!


IOIO Servo Motor Control
IOIO Status : IOIO Not Connect.
Servo Control
: 0 degree
0
90
90
180
Move to control the position:
0
45
90
135
180
Auto Control OFF
Timing +
Timing -

٣ صور الدارات العملية :





٤ ملخص العمل المنجز :

إن مجال العمل مع أنظمة الأندرويد كبير جدا وله استخدامات على كافة الأصعدة حيث أصبح اليوم نظام الأندرويد هو النظام الأكثر انتشاراً على مستوى أنظمة التشغيل الخاصة بالأجهزة المحمولة فضلاً عن شريحة الأردينو التي بدورها تعتبر من أفضل الشرائح في مجال التحكم بسبب البساطة في عملية توصيلها وربطها مع الأجهزة الطرفية وسهولة برمجتها بالنسبة لمختصي التحكم إضافةً لذلك شريحة اليويو المبسطة والتي تختصر الكثير من التعب الذي يستهلكه مبرمج المتحكمات والتي لا تتطلب سوى معرفة احترافية ببرمجة الجافا فهي ملائمة لمختصي الحواسيب المحبين للتحكم عن طريق برمجة التطبيقات .

حاولنا في مشروعنا تغطية بعض ما يمكن إنجازه من خلال ربطهما معا بالإضافة الى إظهار كيفية التعامل مع تقنية الاتصال بلوتوث على المستوى العتادي و البرمجي واظهار الميزات الكبيرة لكل من الأندرويد أو الأردينو والاهتمام البالغ من قبل المبرمجين الهواة والمحترفين بكل منهما

٥ العوائق التي واجهتنا أثناء العمل:

*- مشكلة حجب المواقع الرسمية الخاصة ببرمجة أندرويد مثلا موقع android developer و robotistan و mitchtech, pinterest.

*- مشكلة عدم توفر التجهيزات اللازمة للمشروع

*- ضعف شبكة الانترنت وانقطاعها في بعض الفترات

*- ندرة المواضيع على شبكة الانترنت التي تتحدث عن طريقة الربط بين المتحكمات والأجهزة الذكية

*- صعوبة التعامل البرمجي مع الموارد العتادية الخاصة بالأجهزة الذكية وخطورة الوقوع في الخطأ .

*- صعوبة تعلم لغة الجافا بداية كطلاب تحكم ثم نهاية اكتساب خبرة فيها بعد عناء شديد مرير .

*- مشكلة عدم التوافق التقني بين الكود البرمجي الصحيح وتنفيذه بصحة وبدون أخطاء بالنسبة لأكواد شريحة اليويو بورد رغم صحة كتابته ومنطقيتها ويرجع السبب إلى قلة الموارد المضافة لحزمة بيئة التطوير البرمجي ضمن المنصة eclips بسبب حجبها في المواقع عن سيرفرات بلادنا

بالتالي قمنا كثيراً بعمل وبرمجة أكواد وتطبيقات صحيحة وكاملة ولكن لم يتسنى تنفيذها بسبب الأخطاء التقنية الخارجة عن قدراتنا.

أيضاً :

٥.١ المشاكل والصعوبات

٥.١.١ مشكلة وجود محرك أمامي مستمر لتغير اتجاه الروبوت

كانت الصعوبة في أن المحرك بحاجة لتغذية دائمة للمحافظة على الاتجاه المطلوب حيث أنه يتحرك بمجال محدود وتتم بعدها ممانعته مما يؤدي الى استجرار تيار كبير من البطارية مما يؤدي الى نفادها.

حل المشكلة كان بتوفير بطارية بجهد اكبر وذلك لتخفيض التيار المستجر من المنبع واعطاء قوة دفع أكبر للمحرك الخلفي لزيادة كفاءته في سحب الوزن الكبير للروبوت كبطارية الليثيوم أما الحل الأفضل للمحرك الأمامي كان بتوفير محرك خطوي عوضا عنه و لكن هنا ظهرت مشكلة جديدة كانت بعدم توفر المساحة الكافية لتركيبه بنفس الموقع مما دفعنا للوقوف عند الحل السابق.

٥.١.٢ مشكلة الكود التسلسلي ووجود التأخير الزمني في الكود بالنسبة للمتحكم

كما نعلم أن المتحكمات تعمل بنظام تسلسلي ولا يمكن العمل بالنظام التفرعي وكان وجود أي حلقة تأخير في جزء من الكود سوف تؤدي الى تأخير عمل باقي الكود مما يؤثر سلباً في سرعة استجابة الروبوت للأوامر القادمة من الجهاز الذكي

إنَّ التأخير الزمني في أجزاء الكود هو أمر لا يمكن الاستغناء عنه لذلك كان لابد من البحث عن وسيلة لجعل الكود يعمل بطريقة أشبه بالتفرعية استخدمنا طريقة شبيهة بجدولة أجزاء الكود المتعلقة بالتأخير الزمني في هذه الطريقة تم استخدام عداد الزمن الداخلي للمتحكم من أجل معرفة موعد تنفيذ كل جزء من الأجزاء حيث يوجد في المتحكم عداد زمن يعد الزمن منذ تشغيل المتحكم حتى الوقت الحالي ويمكنه العد لمدة سبعة ايام متواصلة قبل الرجوع إلى قيمة الصفر وهذا يؤمن عمل بسرعة وكفاءة عالية للكود بدون أي تأخير، وبالتالي أصبح لدينا الجزء الذي يحتاج إلى زمن لبدء العمل لن تتم معالجته حتى يصل عداد زمن المتحكم إلى عدد من مضاعفات زمن التأخير المطلوب

٥.١.٣ تأمين نبضات باستمرار للمحرك أثناء عمله

كنا في البداية نستخدم طريقة إرسال نبضات متكررة للأمر الواحد وكان المحرك يعمل لفترة زمنية عند كل نبضة مما أدى لحدوث اضطراب في عمله وأصبح الروبوت يعمل ويتوقف في أزمنة صغيرة حاولنا في البداية زيادة زمن عمل المحرك عند كل نبضة مستقبلة فانخفض الاضطراب قليلا لكن ظهرت مشكلتان الأولى زيادة التأخير الزمني فضلاً عن عدم الاستجابة الفورية للأوامر المرسلة بسبب انشغال المتحكم بتنفيذ النبضات المخزنة سابقا على المحرك أيضاً كانت المشكلة الأخرى بالعدد الكبير للنبضات المرسلة من الهاتف اثناء الضغط على الزر المسؤول عن تشغيل المحرك إلى المتحكم

بعدها كان الحل الأمثل في إرسال أمر واحد من الجهاز الذكي من أجل تشغيل المحرك طيلة زمن الضغط على ذلك الزر ومن ثم إرسال أمر لإيقاف المحرك عندما ينتهي المستخدم من الضغط على الزر ذاته وهذا أدى الى انخفاض عدد كبير من النبضات المرسلة واستجابة فورية من المتحكم

٥.١.٤ خروج الروبوت خارج نطاق الاتصال مع الهاتف الذكي:

كما نعلم أن مشكلة تقنية البلوتوث هي محدودية مجال الاتصال بين الأجهزة وكانت المشكلة لدينا في أن الروبوت عندما يفقد الاتصال مع الجهاز يبقى مستمراً في العمل ولن يستقبل أية أوامر لإيقافه أي أن المحركات لن تتوقف عن العمل إذا كانت في طور العمل قبل فقدان الاتصال مباشرة وهذه مشكلة كبيرة كان الحل أولاً بالاعتماد على حساس المسافة من أجل إيقاف الروبوت قبل اصطدامه ولكن المشكلة في أن سرعة السيارة نسبة إلى سرعة قراءة القيم من الحساس و فحصها وتطبيقها على المحركات كانت كبيرة مما أدى إمكانية إيقافها في نفس المكان وأيضا إذا كان الروبوت يتحرك إلى الخلف لن يتمكن الحساس الموضوع في مقدمة الروبوت من قراءة أي قيمة تم البحث عن الحل أفضل الذي تمثل بقراءة قيم شريحة البلوتوث حيث تحوي كل شريحة بلوتوث على مخرج لتحديد حال الاتصال مع الأجهزة تم الاستفادة من ذلك المخرج و فحصه دوريا من أجل معرفة حالة الاتصال وإرسال أوامر للمحركات بالتوقف إذا فقدنا الاتصال واصدار صوت صفير من مكبر الصوت لتنبيه المستخدم إلى ذلك .

٥.١.٥ إمكانية ارسال قيم حساس المسافة من المتحكم إلى الجهاز الذكي:

المشكلة في البداية كانت في إيجاد مرجع على شبكة الانترنت تتحدث عن طريقة استقبال البيانات عبر البلوتوث من طرف خارجي بالنسبة للجهاز الذكي

ظهرت بعدها أيضا مشكلة استقبال رموز غير مفهومة تارة وعدم استقبال أي بيانات تارة أخرى لم يتم العثور على أي حل للمشكلة التي لم يعرف أصلاً سببها وجدنا في أحد المراجع على شبكة الانترنت فقرة تتحدث عن معدل الترميز بين بين المرسل والمستقبل ووجوب التنسيق بينهما فقد كان المتحكم لا يستقبل البيانات من الجهاز الذكي لأن معدل ترميز بيانات شريحة البلوتوث أكبر بالتالي وافقنا بين معدل الإرسال والاستقبال Baud rate وبذلك حُلت المشكلة .

٥.١.٦ مشكلة انهيار البرنامج عند التشغيل لأول مرة:

كانت هذه المشكلة إحدى أكثر المشاكل صعوبة وذلك بسبب عدم معرفة السبب الرئيسي الذي يؤدي إلى انهيار البرنامج عندما يتم تشغيل التطبيق لأول مرة عندما لا يكون البلوتوث مفتوحاً مسبقاً وذلك لأنه يتم حجب الخيط الرئيسي المسؤول عن البرنامج الذي يدعى ui thread من قبل نظام الأندرويد بسبب قيامه بتشغيل البلوتوث مما يؤدي ال قيام النظام بإغلاق التطبيق

حل تلك المشكلة كان باستحداث خيط جديد مهمته تشغيل البلوتوث واتمام عملية الاتصال مع شريحة البلوتوث بشكل منفصل عن الخيط الرئيسي ومن ثم انهاء هذا الخيط بسبب انتهاء مهمته أيضاً تم استحداث خيط ثالث في التطبيق مهمته استقبال الرسائل الواردة من المتحكم وارسال الأوامر اليه كي نحافظ على تطبيق متوازن وخالي من الأخطاء حيث أن هذا الخيط يبقى دائما بحالة انتظار للرسائل الواردة كي لا يشعر المستخدم أن هناك بطء في عمل التطبيق

٥.١.٧ مشكلة محدودية عرض حزمة قناة الاتصال:

كانت المشكلة متمثلة أنه يوجد تأخير في استجابة النظام للأوامر المرسلة بسبب وجود القراءات المرسلة من الحساس عبر المتحكم مما أثر سلباً على سرعة أداء النظام بكامله حيث كنا نقوم بإرسال عدد كبير من القراءات في الثانية الواحدة مما يؤدي الى اشغال قناة الاتصال بإرسال تلك القيم

تم حل هذه المشكلة بتخفيض عدد القراءات المرسلة في الثانية الواحدة من المتحكم الى الهاتف

تم ذلك عن طريق زيادة حلقة التأخير الزمني بين كل عملية إرسال ولكن بقي هنالك جزء من المشكلة بأنه يتم وضع النظام في حالة سكون اثناء حلقة التأخير ولن يستجيب للنبضات المستقبلة خلال فترة سكونه مما كان له تأثير كبير على أداء النظام

تم الحل أيضا كما ذكرنا سابقا عن طريق جدولة أجزاء الكود زمنياً .

٥.١.٨ مشكلة المحافظة على الاتصال أثناء عملية التبديل بين أنماط التحكم:

في البداية كان تطبيق الأندرويد عبارة عن واجهة واحدة حيث كان هناك object واحد مسؤول عن عملية الاتصال لذلك لم تظهر في البداية أية مشاكل ظهرت مشاكل الاتصال بشكل جلي عند القيام ببرمجة التطبيق ليدعم أكثر من نمط للتحكم (تحكم بحساسات التسارع الزاوي بالموبايل، تحكم بالصوت ، تحكم بأزرار التطبيق ، تحكم بجويستيك التطبيق) مما أدى للحاجة لعمل واجهة لكل نمط من تلك الأنماط المشكلة تمثلت بأن كل واجهة عند الانتقال منها الى أي واجهة أخرى تقوم بإنهاء عملية الاتصال وكان من الواجب أن تقوم كل واجهة بإعادة تمكين الاتصال لم يكن ذلك عمليا بسبب الوقت المهدور خلال عملية إعادة الاتصال مما يكون سبباً في عرقلة المستخدم قمنا بحل هذه المشكلة في البداية عن طريق جعل التطبيق عبارة عن واجهة واحدة تتضمن جميع أنماط التحكم حيث يتم الانتقال بين تلك الأنماط عن طريق اخفاء العناصر الغير مطلوبة في النمط الجديد واظهار العناصر المطلوبة فيه ولكن واجهنا في هذا الحل صعوبة برمجية تمثلت باضطرارنا لكتابة كود خاص للواجهة بلغة الجافا بالإضافة الى وجود بطء في البرنامج

الحل الأمثل كان بالعودة الى نظام الواجهات المتعددة والقيام بحجز ال object المسؤول عن عملية الاتصال من نوع static حيث يعبر هذا النوع عن أن هذا ال object هو وحيد على مستوى كامل البرنامج ولن يتم حجبه إلا بانتهاء البرنامج كاملاً.

استطعنا من خلال هذه الطريقة الحفاظ على اتصال واحد يتم استخدامه من قبل جميع الواجهات.

٦ الخبرات العلمية والعملية:

١- خبرة كبيرة في مجال التعامل مع المتحكمات لوحات التطوير(أردينو).

٢- خبرة كبيرة في مجال برمجة الأندرويد للقيام بأعمال التحكم.

٣- تعرفنا على عناصر إلكترونية جديدة وقمنا باستخدامها والاستفادة من مميزاتها .

٤- قمنا بدراسة لغات برمجة جديدة وواسعة الانتشار خاصة الجافا والجافا غرافيكس .

٥-قمنا في هذا العمل بربط الأندرويد بالمتحكم أردينو والشريحة IOIO/OTG .

٧ آفاق تطوير المشروع والصعوبات:

١- من خلال هذا المشروع وبعد تخطي الصعوبات في مجال الربط والبرمجة أصبح بالإمكان التعامل مع أي نوع من أنواع المتحكمات باستخدام نفس طريقة التواصل.

٢- بالتعرف على بورد اليويو والتي تعد مجهولة في بلادنا والتي قد أدخلناها الآن لتحفل بها مكتبة الكلية وبعد قمنا بوضع ركيزة أساسية لتعلمها والتدرب عليها وتركنا ملحقات ملفات خاصة بها للتدرب عليها وإمكانية استغلالها في تطوير المشروع بشكل عام ليرقى إلى مستويات ضخمة نطمح لها وعلى سبيل الوصف لا الحصر :

التحكم من شاشات الأندرويد بأجهزة المتحكمات المنطقية القابلة للبرمجة PLC عن طريق وصلها وتخاطب معالجها مع شريحة اليويو(وهو ما أطمح له) .

صناعة آلات CNC بتطبيقات تحكم برمجية رسومية مباشرة عن طريق الأجهزة باليويو.

٣- إمكانية استخدام تقنيات اتصال أخرى ذات مدى أبعد WIFI/3G/GPS

٤- إمكانية التحكم بأي نوع من أنواع الروبوتات أو التحكم بالمنزل أو حتى الآلات في المصانع وبناء أنظمة مراقبة على الهاتف الذكي

أخيراً :

الاستفادة من المشروع ليكون أرضية تبنى عليها مشاريع مستقبلية في :

المجالات طبية : تحكم بكراسي العجزة والنقالات بالأندرويد.

المجالات جوية: طائرات بدون طيار تتحكم بها الأجهزة الذكية .

مجالات رفاهية : قيادة السيارة وأنت متكئ عن طريق جهازك الأندرويد.

المجالات بحرية : غواصات روبوتية .

المجالات صناعية : مراقبة وتحكم بالأذرع الروبوتية الإنتاجية .

نأمل في نهاية هذا العمل أن نكون قد وفقنا في ايصال الفكرة المرجوة من هذا العمل وأن نكون قد قمنا بتغطية كافة الجوانب التي سعينا إليها في هذا المشروع والله من وراء القصد.

وآخر دعوانا أن الحمد لله رب العالمين .



المراجع:

كتاب :



Making Android Accessories with ioio By Simon Monk_ ٢٠١٢م

abdullah ali abdullah simply arduino [كتاب]. - ٢٠١٢.

Brian Evans Begining Arduino Programming [كتاب]. - ٢٠١١.

Reto Meier Android 4 Development [كتاب]. - ٢٠١٢.

Reto Meier Android Application Development [كتاب]. - ٢٠١٣.

قناة المبرمج التايلاندي تاي الملقب ماستر ينج :

courses of Thai master UNG

https://www.youtube.com/user/easy4com/videos?view=0&shelf_id=1&sort=dd

http://www.theengineeringprojects.com

موقع تعرف http://www.arduinopro.com

موقع العربي www.Arabtronix.net

موقع شركة SparkFun

http://www.firmcodes.com

courses of Viviana Ruiz

دروس الأستاذ عبد الله عيد في جافا الأندرويد وجافا١٠١و١٠٢و١٠٣على قناته اليوتوب

موقع https://github.com/ytai/ioio/wiki

كتاب: ٢٠١٣- Professional Android Open Accessory Programming with Arduino

كتاب : ٢٠١٢ Beginning Android ADK with Arduino-

كتاب: ٢٠١١ Arduino Robotics-

كتاب : Simply Raspberry Pi لـ**abdullah ali abdullah**

http://www.cisco.com

http://www.ieee.org

http://en.wikipedia.org

http://www.android.com

http://www.sparkfun.com

http://www.atmel.com/literature

http://www.microchip.com/

هندسة الروبوت : أ. د . الياس طوشان

محاضرات في التحكم المتقدم : د . أحمد أكراد

محاضرات في الإلكترونيات الصناعية : د . عماد الروح

فن الإلكترونيات (الدارات التماثلية) : باول هورويتز – وينفيلد هيل

****-----*******-----*******-----*******-----*******-----*******-----

ترقبوا في السنة والنصف القادمة جزءاً ثاني من هذا الكتاب نتناول به البرمجة لشريحة IOIO بشكل أعمق وأكثر احترافية وتطوراً لنربطها مع المتحكمات الأخرى ونبني تطبيقات قادرة على ارسال أوامر التحكم واستقبال القراءات عن طريق شبكات الإنترنت والوايفاي ولأجل التفصيل والشرح من البداية حتى الاحتراف في هذا المجال تابعونا بالمستقبل على قناتنا باليوتيوب التي سنتناول بها العديد من المواضيع منها كورس كامل تدريبي بهذا المجال ما تيسَّر لنا ذلك مستقبلاً.

حلب ٢٠١٦/٧/٢١م

الموافق ١٦/شوال/ ١٤٣٧هـ

قناتنا باليوتيوب https://www.youtube.com/channel/UCq0RbRuLe3oFb9e12vqzaUg

النهاية

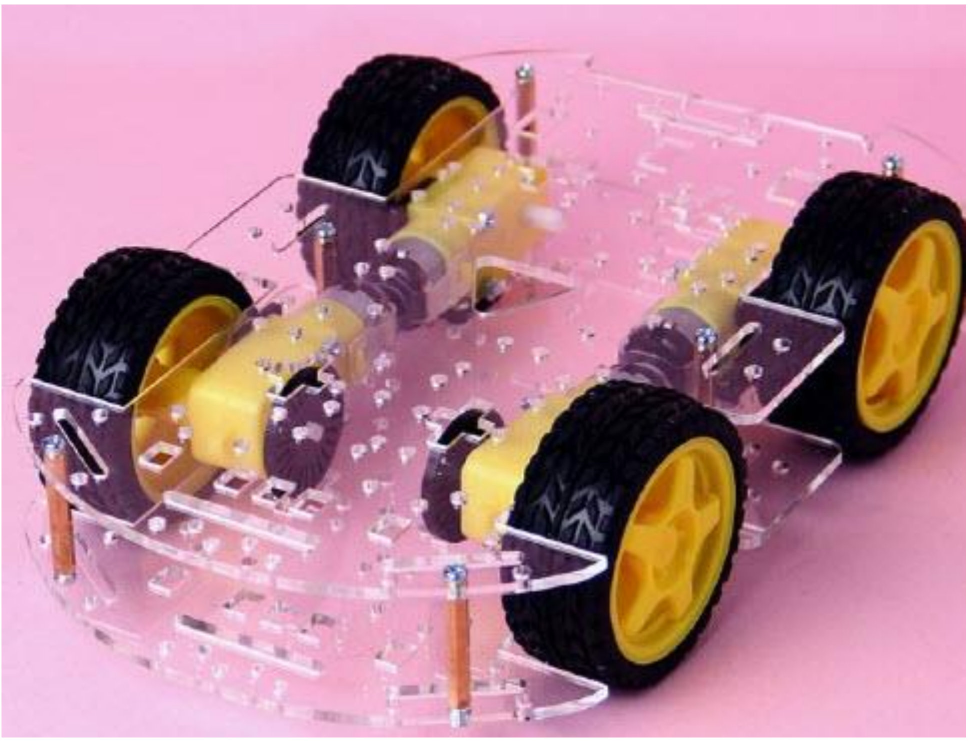
بعض شرائح القيادة الهامة للروبوتات السيَّارة

هياكل مميزة

انكودرات حساسات سرعة

المحركات الخطوية

Smart Car Chassis Model: KYS002



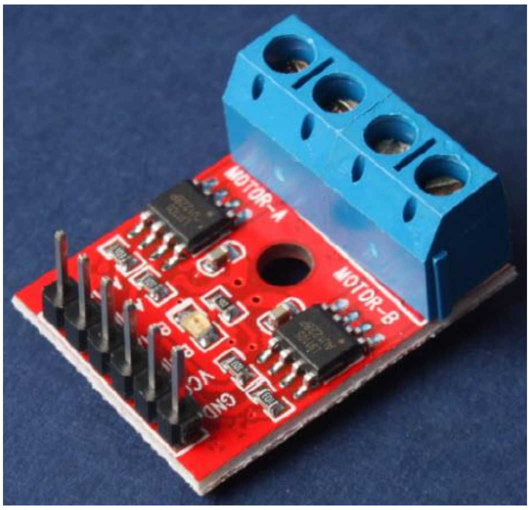
بأربع محركات مستمرة أقصى جهد تتحمله ١٤فولط سرعة تهييج متغيرة بنسبة 1/48 بحجم قياس أبعاده للفجوات

4x Car Tire (size: about 6.5 x 6.5 x 2.6 cm)

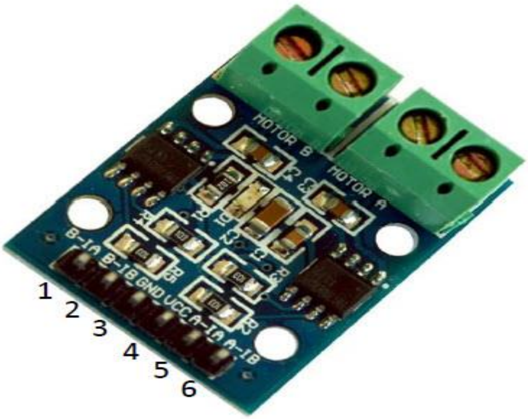
جدول خصائصها ومزاياها

Specifications:

Operating Voltage	3 - 9V
Current	600MA
Reduction rate	1:48
Thicker	3mm
Driving type	4WD Robot Chassis
Weight	2kg
Car Chassis Size	24.5 x 15.5 cm

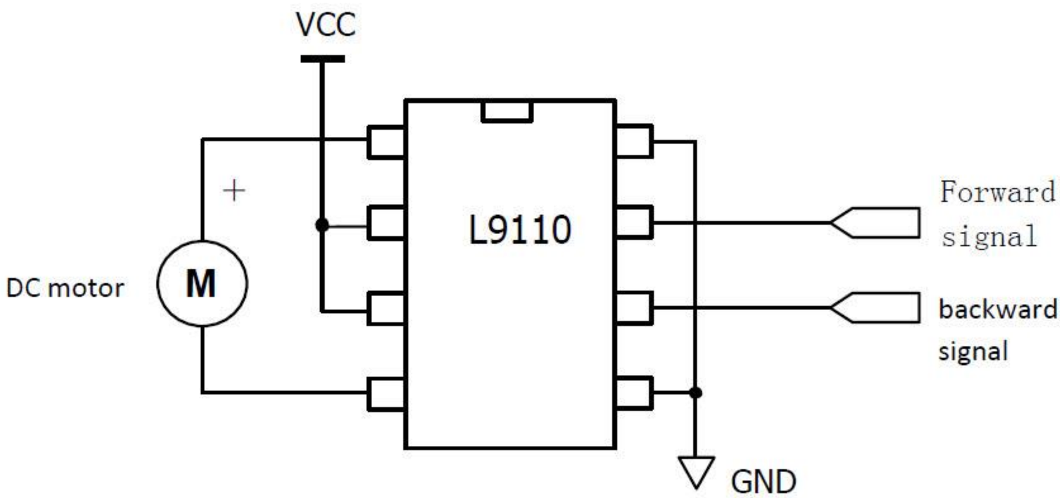


موديول القيادة لمحركين مستمرين



1. B-IA: Motor B Input A
2. B-IB: Motor B Input B
3. GND: ground
4. VCC: 2.5V-12V DC
5. A-IA: Motor A Input A
6. A-IB: Motor A Input B

تحوي اثنتين من الدارة المتكاملة L9110



من أهم مواصفاتها أن أقصى خرج على مخارجها بقيمة لا تتجاوز القيمة الأعظمية ١٢ فولط غير ذلك سيحصل قصر بالدارة المتكاملة وبالتالية قصر لمنافذ الدخل للموديول وبالتالي قصر لأي منفذ قادم من شريحة متحكم مبرمج إليهما فيفضل للحماية وضع ديود بينهما

built-in clamp diode reverse the impact of the current

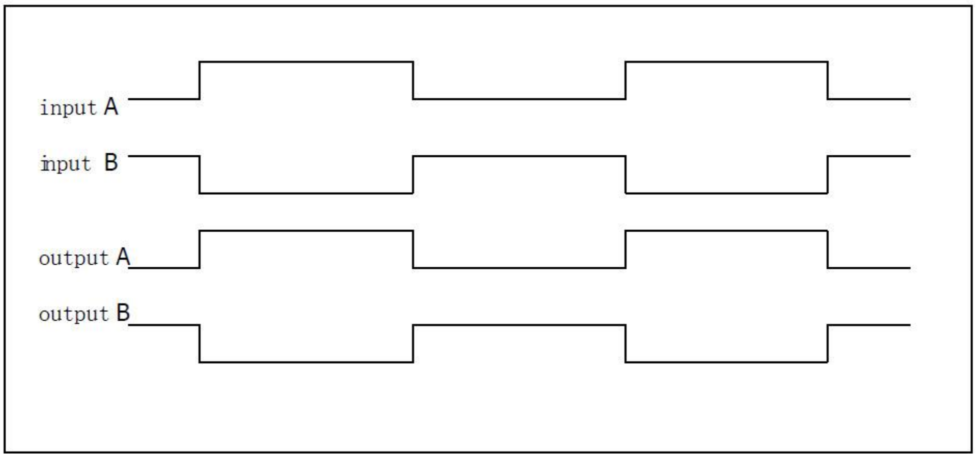
وأيضاً الخرج لا يعطي أكثر من ٨٠٠ مل أمبير كحد أقصى

أفضل استخدام هو جهد تغذية بقيمة ٩ فولط وتيار استجرار ٧٥٠ مل أمبير

ترانزستورات منافذها التحكم قابلة للتعديل العرضي المطالي بتردد حتى ٧٥ هرتز بنسبة ١٠٠% على جهد قدح ٤.٥ فولط وأقل أما أكثر من ذلك فتقل النسبة .

جداول خصائصها المميزة وأشكال إشاراتها البيانية:

Pin waveform diagram:



Electrical characteristics:

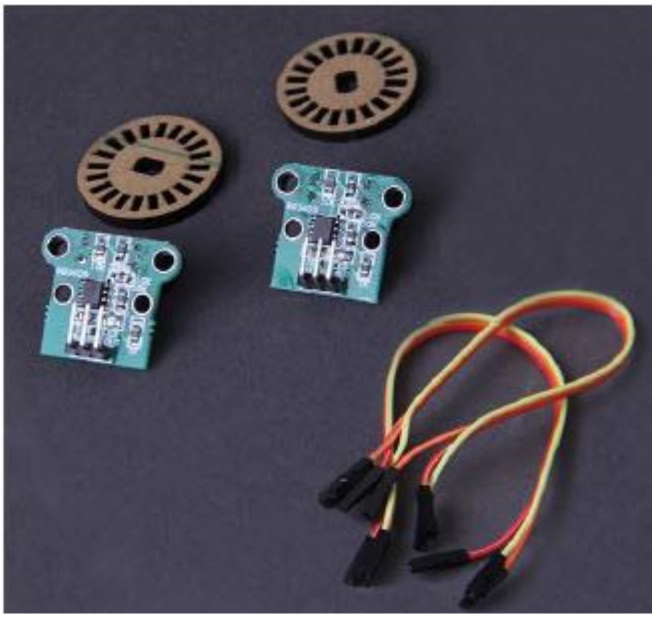
Symbol	Parameters	Range			Units
		Min	Typical	Max	
VCC	Supply Voltage	2.5	6	12	V
Idd	Quiescent Current	—	0	2	uA
Iin	Operating current	200	350	500	uA
IC	Continuous	750	800	850	mA
IMax	Current peak	—	1500	2000	mA

Logical relationship:

IA	IB	OA	OB
H	L	H	L
L	H	L	H
L	L	L	L
H	H	L	L

حساس السرعة الإنكودر

Double Speed Measuring Module with Photoelectric Encoders Model: HC-020K



مزاياه

Specifications:

1. Module Working Voltage: 4.5-5.5V

2. Launch Tube Pressure Drop: VF=1.6V

3. Launch Tube Current: If<20mA

4. Signal output: A, B two lines; TT power level;

5. Resolution: 0.01mm

6. Measurement frequency: 100 KHz

7. Color: Green + black

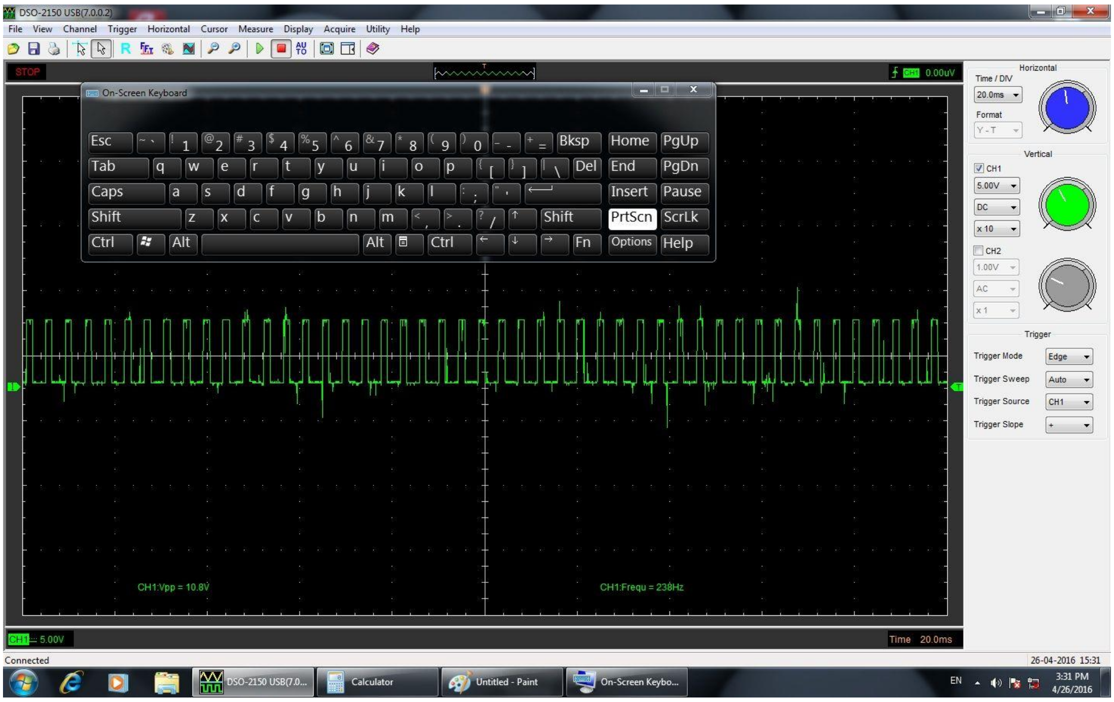
8. Disc diameter: 24mm

9. Inner Disc Diameter: 4mm

10. Encoder resolution: 20 lines

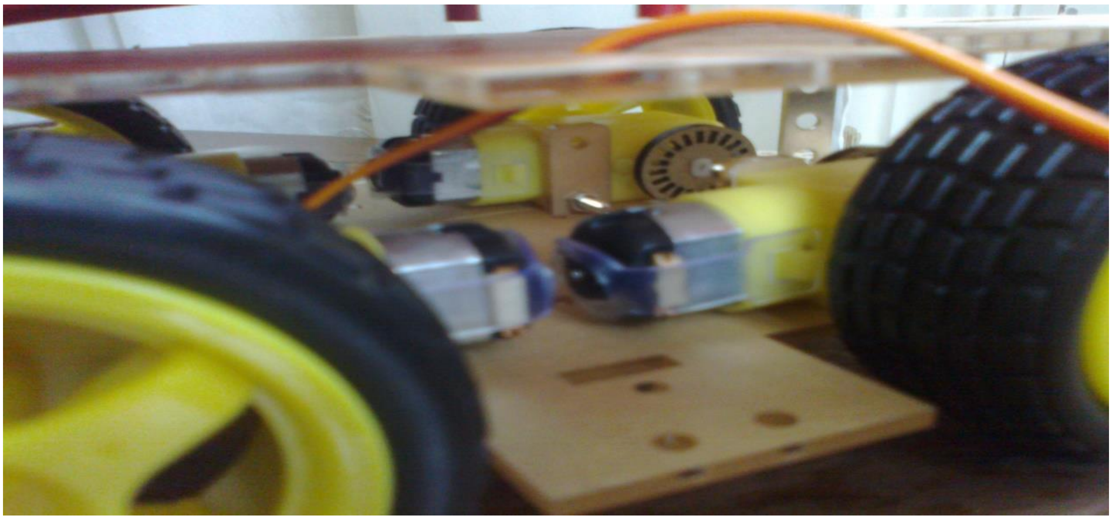
11. Speed measuring sensor configuration: measure line 1 motor speed 12. Application: For experiment

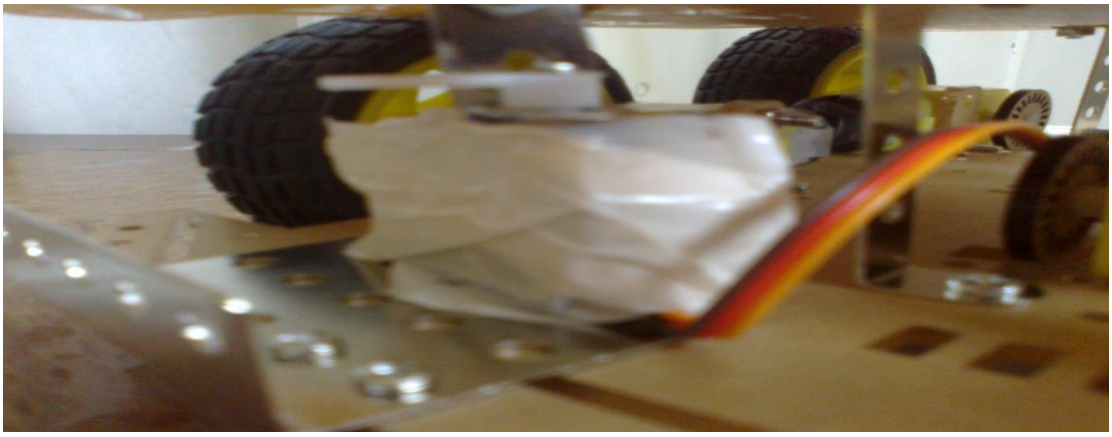
بعد القياس والتجريب تبين ٢٤٠ هرتز خلال ثانية، ١٠ فولط من القمة للقمة ولدينا ٢٠ ثقب أي ٢٠ نبضة on بالدورة الواحدةبالتالي السرعة : ٧٢٠دورة بالدقيقة



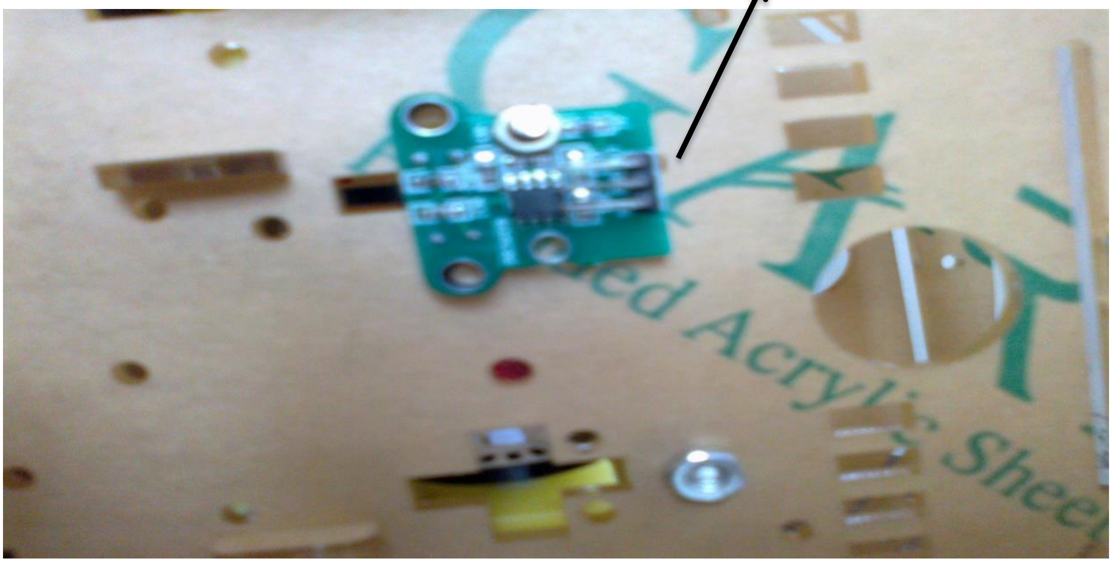
لمن أحب استخدام حساسات تيار لروبوتات سيَّارة ننصح بحساسات ACS712

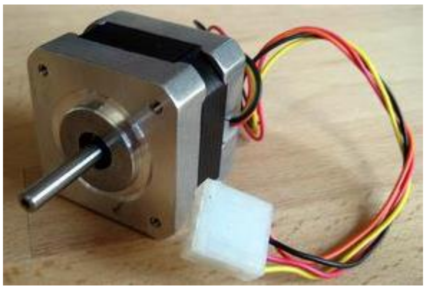






الإنكودر على الهيكل





محركات الخطوة : يطلق عليها أيضاَ المحركات متدرجة الحركة

بدأ التفكير في استخدامها عندما بدأ الطلب على جهاز يمكن أن يعطي حركة دورانية في صورة انحراف زاوي ثابت . عملياَ يمكن ربط المحرك الخطوي مع الحاسب الآلي أو المعالجات الصغيرة التي تستطيع التحكم به لأداء وظائف معينة فمحركات الخطوة (stepper) هي محركات يتم التحكم بها رقميًا وتدور عددًا محددًا من الدرجات (خطوة) في كل مرة – **فيمكن تحريك شيء ما عبر هذه المحركات لتقف بموضع محدد** وبالتالي تحدد دقة مطلوبة للتوقف في الموضع لا يحققها المحرك المستمر لعطالته ودورانه المستمر إذ ليس له سوى حالتي ١/٠---**on/off**

تدعى أصغر الدورانات بخطوة المحرك فمثلاً محرك ذو خطوة ٠.٧ درجة يحقق ٥١٤.٢ خطوة كل دورة ٣٦٠(درجة)

تطبق نبضة clock على دارة خاصة تستخدم للتحكم بمحرك الخطوة .إن عدد الدرجات في الخطوة الواحدة لمحرك الخطوة يمكن أن يكون صغيرًا إلى حد 0.72 درجة في الخطوة الواحدة أو كبيرًا ويصل إلى 90 ° في الخطوة .في المحركات العامة التي تستخدم في تطبيقات متعددة الأغراض يتراوح عدد الدرجات في الخطوة بين (15-30) درجة . يدور محرك الخطوة، بعكس السيرفو 360 درجة ويمكن جعله يدور بشكل دائم مثل محرك DC (ولكن سرعة دورانه الأعظمية dc أقل من سرعة محرك) تؤمن محركات الخطوة عزمًا كبيرًا عند سرعات دوران منخفضة مما يجعل هذه المحركات مفيدة جدًا في التحكم بالموضع بدقة عالية وسرعات منخفضة . تستخدم محركات الخطوة في الطابعات للتحكم بتغذية الطابعة بالورق كما تستخدم في التيليسكوبات التي تلاحق النجوم .تستخدم محركات الخطوة أيضًا في الراسمات (Plotters) و في تطبيقات أخرى عديدة منها تحديد الموضع أو الدوران إلى موضع باستخدام الحساسات.وكما في تجربتنا هنا نظام الرافعة الحاملة الملتقطة ضمن السيور الناقل . ومنه فإن مميزات المحركات الخطوية :

عزم المسك : فالمحرك الخطوي يحافظ على موضعه بثبات عندما لا يكون في حالة دوران حتى وإن كانت هنالك حمولة مطبقة عليه . **سرعة الدوران**: تتعلق بسرعة إشارات الدخل ومستقلة عن الجهد على عكس المحركات المستمرة التي يتم التحكم بسرعتها عن طريق تغيير جهد التغذية (تحكم محدود يحتاج جهد أصغري ليبدأ الدوران وهي طريقة تؤثر على العزم) **تمتلك عزماً كبيراً** عند السرعات المنخفضة **التحكم بجهة الدوران** يتم بعكس تسلسل إشارات الدخل .

قراءة مواصفات تشغيل المحرك الخطوي :

<u>جهد مستمر</u> الإشارة يعمل عليه المحرك

<u>تيار الملف هو التيار الأعظمي</u> لأن مقاومة الملف منخفضة ١٠٠أوم لكل ملف وممانعتها من ١٥٠-٢٥٠ميلي هنري

<u>الخطوة</u> تقاس بالدرجات قد نصادف محركات كتب عليها عدد الخطوات بالدورة الواحدة عندها : Step = 360/n

التردد الأعظمي : هو تردد الإشارة التي تحدد الانتقال بين الخطوات للتحكم بسرعة دوران محور المحرك وهذا التردد لا يجب تجاوزه ف ٢٠٠hz -٣٠٠ hz تعني ٢٠٠-٣٠٠ rpm .

عزم المسك : بما أن المحرك الخطوي مصمم للتحريك بدقة لموضع محدد فمن الضروري الأخذ بعين الاعتبار مسؤولية تحمل محور المحرك للحمولات الزائدة عندما لا يكون محوره بحالة دوران فتحدد الشركة الصانعة العزم الكافي لقيادة الأجسام مع المحافظة على ثبات الزاوية بقيمة تعطى بالميللي نيوتن _ متر m-mN

أ نواع محركات الخطوة :

يوجد نوعان رئيسيان للمحركات الخطوية هما :

- **المحركات ذات الممانعة المتغيرة**
- **المحركات ذات المغانط الدائمة**

ويوجد نوع ثالث هجين بين النوعين السابقين يدعى العمومي.

وبغض النظر عن الفروقات بين أنواع المحركات إلا أن النتيجة المطلوبة هي الدوران بخطوة واحدة فقط أي دوران المحور بزاوية محددة مع كل نبضة يتلقاها أحد ملفات المحرك وتتغير هذه الزاوية تبعاَ لبنية المحرك الداخلية .

ولكنها محصورة عموماَ بين ٠.٩ درجة حتى ٩٠ درجة .

المحركات الأكثر استعمالاً هي ذات خطوة :

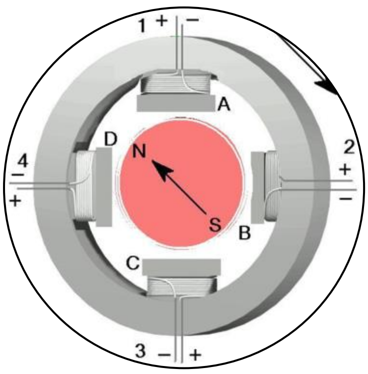
- ٠.٩ درجة أي ٤٠٠ خطوة في الدورة .
- ١.٨ درجة أي ٢٠٠ خطوة في الدورة .
- ٣.٦ درجة أي ١٠٠ خطوة في الدورة .
- ٧.٥ درجة أي ٤٨ خطوة في الدورة .
- ١٥ درجة أي ٢٤ خطوة في الدورة .

من الواضح أن المحركات الخطوية ذات دقة مرتفعة جداَ وعمر طويل جداَ وذلك لان التآكل الميكانيكي غير موجود عملياً (لا يوجد احتكاك) . يفضل استخدام هذه المحركات في المجالات التي تتطلب دقة عالية ومنها المكونات الميكانيكية المستخدمة في مجال المعلوماتية والروبوت .

تتوفر المحركات الخطوية بعدة قياسات لمحاورها بدءا َمن ١ سم وحتى حوالي ١٠ سم

لاحظ أنه يتم التعبير عن العزم بواحدة الكيلو غرام في السنتمتر kg / cm

وهذا ما يحدد الثقل بواحدة الكيلو غرام الذي يستطيع محور المحرك (ذو الطول المقدر بالسنتمتر) أن يتحمله .



يتغير جهد تغذية المحرك كثيراَ تبعاَ لنوعه فقد يكون محصوراَ ضمن مجال من ٣ فولت وحتى عشرات الفولت .

أيضاَ وتبعا للمقاومة الخاصة بملفات المحرك ، يختلف التيار المستهلك في المحرك بدءاَ من عشرات الميللي أمبير وحتى الكثير من الأمبيرات . وببساطة يمكننا القول أنه كلما كان التيار المستهلك كبيراَ كان العزم أكبر .

أ - المحركات ذات المغانط الدائمة

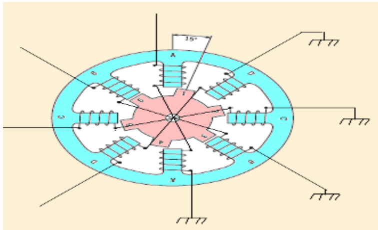
هذا النوع يقسم بدوره إل نوعين :

*** محركات أحادية القطبية .**

*** محركات ثنائية القطبية**

المحركات ذات المغانط الدائمة تتكون من ثابت يحوي ملفات و دوار مغناطيسي (مغناطيس) وتمتاز هذه المحركات بأن زواياها الخطوية كبيرة نسبياً من (٧.٥ – ١٥) درجة . ولكن ممزه العزم السرعة جيدة .

يبين الشكل البيانية المبسطة لمحرك خطوي ذو مغناطيس دائم

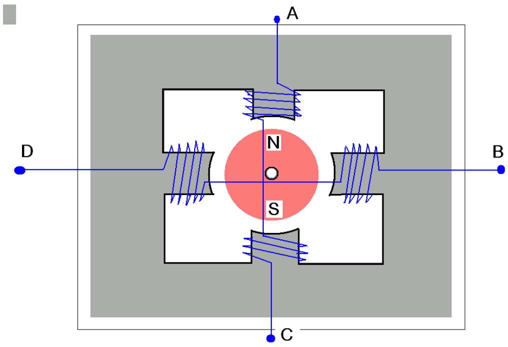


١ – المحركات أحادية القطبية :

يبين الشكل المخطط البينوي للمحرك أحادي القطبية حيث نلاحظ أن كل قطب من أقطاب الثابت لف عليه ملف مستقل) طرفاه مستقلين عن طرفي أي قطب آخر (أي لكل ملف سلكين حيث يتم وصل أحدهما للقطب الموجب (+ من التغذية والآخر بالقطب السالب (-)

من منبع التغذية ونلاحظ ببساطة أنه لعكس جهة التيار نعكس توصيل السلكين بالموجب والسالب . يعد المتحكم بهذا النوع من المحركات الأسهل مقارنة بالنوعيات الأخرى . إذ يكفي تغذية الملفات بالتتالي لتدوير محور المحرك . كما هو مبين في الشكل

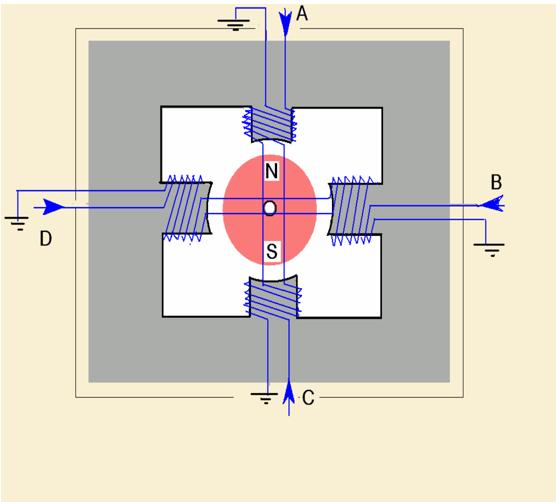
٢ – المحركات ثنائية القطبية :



يبين الشكل البنية الداخلية لمحرك ثنائي القطبية حيث في الملف على قطبين متقابلين ويصدر من كل قطب طرف واحد للملف (سلك واحد) وتتطلب هذه النوعية من المحركات آلية قيادة أكثر تعقيداً مقارنة بالنوع احادي القطبية إذ يجب

تغيير جهة التيار المار في الملفات بعد كل خطوة دورانية

ب - المحركات ذات الممانعة المغناطيسية المتغيرة



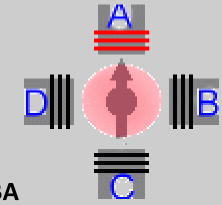
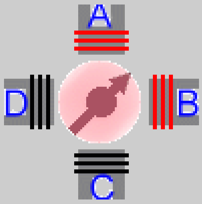
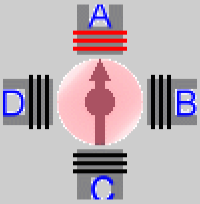
تعمل محركات الخطوة ذات الممانعة المغناطيسية المتغيرة وفق نمط يختلف عن ذلك الخاص بمحركات المغانط الدائمة . فهي تحوي بالطبع على ثابت ودوار ولكن الدوار هنا مصنوع من الفولاذ النقل غير الممغنط ، وهو ليس أملساً حيث أنه يحوي العديد من الأسنان ، يبين الشكل هذه النوعية من المحركات حيث نجد في هذا الشكل أن الثابت مكون من ٨ نضوات فل يتم لف الملفات عليها و هذا مايعطي أربعة أطوار أما الدوار فيحوي على ٦ أسنان أي (٦) أطوار

جـ محركات الخطوة العمومية :

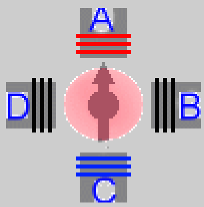
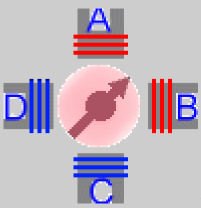
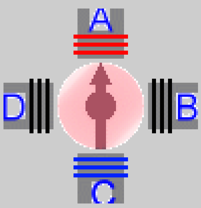
تمثل هذه المحركات الخطوية نموذجًا هجينًا من المحركات وحيدة وثنائية القطبية. للمحرك العمومي أربعة ملفات مستقلة وثمانية أسلاك. بوصل الملفات على التوازي فإن المحرك العمومي يتحول إلى محرك خطوي وحيد القطبية، أما إذا وصلنا الملفات على التسلسل فإن المحرك يتحول إلى محرك خطوي ثنائي القطبية.

آلية التحكم بالمحرك أحادي القطبية

هناك ثلاث طرق رئيسية لقيادة جميع أنواع المحركات الخطوية وهي :

الخطوة الكاملة	الخطوة المزدوجة	نصف الخطوة
 DCBA	 DCBA	 DCBA
000A	00BA	000A
00B0	0CB0	00BA
0C00	DC00	00B0
D000	D00A	0CB0
		0C00
		DC00
		D000
		D00A

<u>آلية التحكم بالمحرك ثنائى القطبية</u>

الخطوة الكاملة	الخطوة المزدوجة	نصف الخطوة
		
AC	AC-BD	AC
BD	BD-CA	AC-BD
CA	CA-DB	BD
DB	DB-AC	BD-CA

CA	AC-BD	AC
CA-DB	--	--
DB	--	--
DB-AC	--	
AC		
--		

في نمط الخطوة الكاملة يكون العزم صغيراً لأنه تتم تغذية ملف واحد فقط لتحقيق دوران المحرك بخطوة كاملة . أما في نمط الخطوة المزدوجة يعطي المحرك اكبر استطاعة ممكنة (عزم مرتفع) لأنه تتم تغذية ملفين معاً في نفس اللحظة .

نلاحظ ان نمط التحكم بنصف خطوة يتيح مضاعفة عدد الخطوات التي يستطيع ان يحققها المحرك خلال دورة واحدة

ولكن هذا ما يسبب عزم دوران غير منتظم . حيث تمثل قيادة المحرك وفق هذا النمط مزيجاً بين نمطي الخطوة الكاملة و الخطوة المزدوجة ففي هذا النمط إذا كان الحمل كبيراً قد يتم تجاوز بعض الخطوات وهذا غير مرغوب طبعاً .

	PHASE			
	A	B	$\bar{A}$	$\bar{B}$
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1

One-Phase On, Full Step

Figure 5a.

	PHASE			
	A	B	$\bar{A}$	$\bar{B}$
1	1	1	0	0
2	0	1	1	0
3	0	0	1	1
4	1	0	0	1

Two-Phase On, Full Step

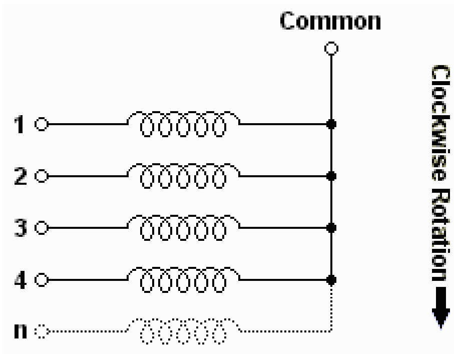
Figure 5b.

	PHASE			
	A	B	$\bar{A}$	$\bar{B}$
1	1	0	0	0
2	1	1	0	0
3	0	1	0	0
4	0	1	1	0
5	0	0	1	0
6	0	0	1	1
7	0	0	0	1
8	1	0	0	1

آلية التحكم بالمحركات الخطوية ذات الممانعة المغناطيسية المتغيرة :

تتم قيادة هذه المحركات كما في المحركات السابقة بالطرق الثلاث (خطوة كاملة ، خطوة مزدوجة ، نصف خطوة)

وهي من أسهل المحركات التي يتم التحكم بها تتألف من مجموعة ملفات عددها nويزيد عددها مع دقة خطوة المحرك ويبين الشكل التالي البنية وتتالي التغذية :



Index	1	2	3	4
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1
5	1	0	0	0
6	0	1	0	0
7	0	0	1	0
8	0	0	0	1

الجدول يبين تتابع الأوامر لقيادة هذا المحرك :

نلاحظ أن قيادة هذا النوع مطابق لطريقة قيادة المحرك ذي المغناطيس الدائم أحادي القطبية وذلك لأن احد طرفي كل طور موصول موصول مع القطب الموجب (+) أو مع القطب السالب (-) من التغذية وبالتالي ليس هناك حاجة لعكس التيار ضمن الطور .

الجدير بالملاحظة أن هناك فرق بين عدد أقطاب الثابت وعدد أسنان الدوار حيث يصنع الدوار بعدد أسنان أقل من عدد أقطاب الثابت (يكون عادة أقل بسنين) وذلك للاستفادة من الفروق الزاوية بينهما : ففي المثال الشكل السابق

(١.٧) نلاحظ أن عدد أقطاب الثابت = ٨ أقطاب و عدد أسنان الدوار = ٦ أسنان :

العدد ٨ يمثل بالزوايا (٣٦٠ / ٨ = ٤٥) ، والعدد ٦ يمثل بالزوايا (٣٦٠ / ٦ = ٦٠)

والفرق بين الزاويتين هو ٦٠ – ٤٥ = ١٥ درجة . أي أن هذا المحرك يتحرك بزاوية قدرها ١٥ درجة .

ومن أجل تغيير زاوية الخطوة فإننا نغير من عدد الملفات أوعدد الأسنان

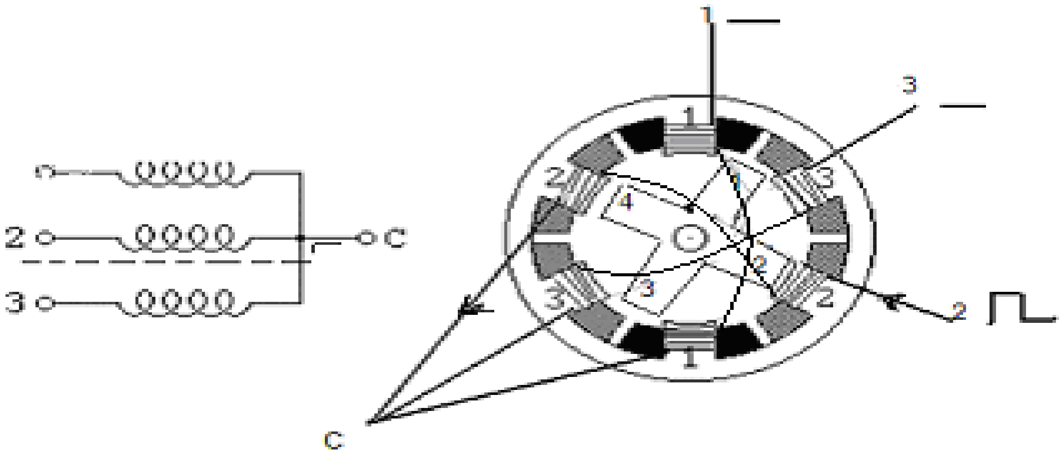
(زاوية الخطوة = ٣٦٠ / عدد أزواج أقطاب الثابت × عدد أسنان الدوار)

الخطوة الكاملة	الخطوة المزدوجة	نصف الخطوة
1	1+2	1
2	2+3	1+2
3	3+1	3
	--	2+3

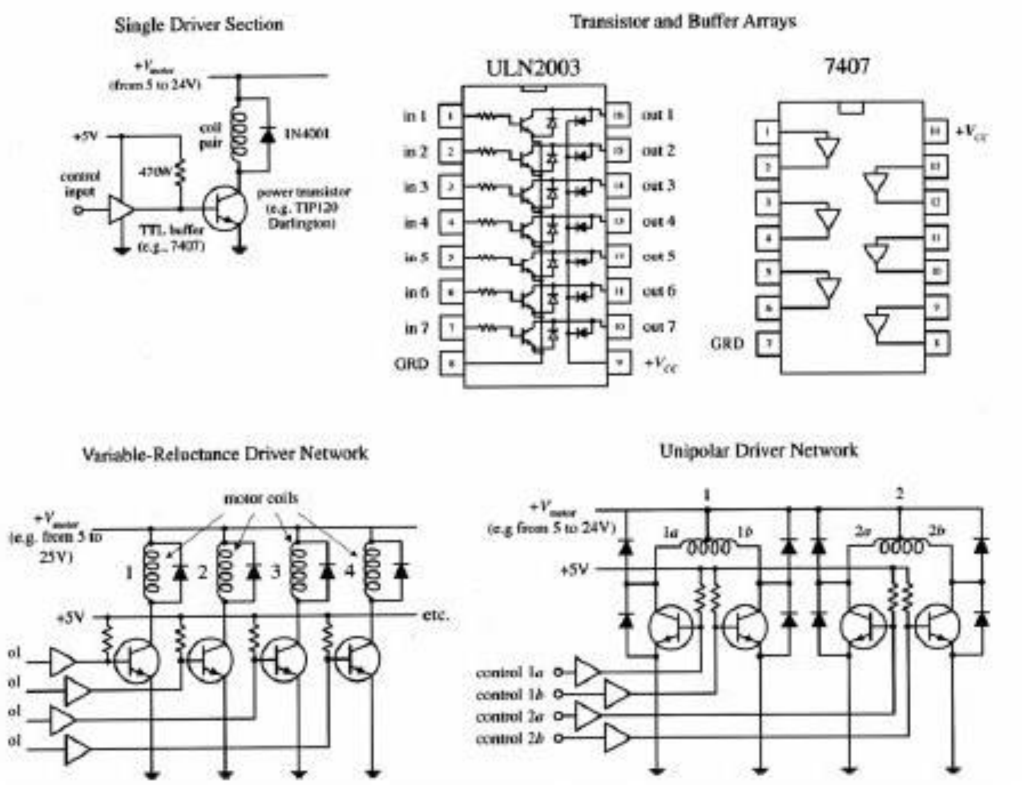
3	--	-
3+1	--	-
--	--	-
--	--	

هذا المحرك له (٦) أقطاب على الثابت

النقطة مشتركة إلى الأرضيCو (٤) أسنان على الدوار



قيادة المحركات الخطوية : يحتاج كل محرك خطوة إلـى دارة قيادة تستطيع التحكم بتدفق التيـار الذي يمر عبر ملفات القسم الثابت في المحـرك، ودارة القيـادة بـدورها يجـب أن تقـاد بواسطة دارة منطقيـة تسـمى translator إذا كنت لا ترغب باستخدام العناصر المنفصلة فبإمكانك استخدام



دارات متكاملـة IC لمصـفوفات ترانزسـتورية، مثـل سلسـلة ULN200X مـن شـركة Allegro Microsystems , او سلسـلةDS200X مـن شـركة National Semiconductors. الـدارة المتكاملـة ULN 2003 المبينـة فـي الشـكل (21) هـي شـريحة متآلفـة مـع TTL وتحـوي سـبعة ترانزستورات دارلنغتون مع ديودات حماية داخلية .يمكن استخدام دارة العوازل المتكاملـة 7404 مع الدارة ULN 2003 لتشكيل دارة قيـادة كاملـة لمحرك خطـوي أيضـاً .تستطيع الـدارة المتكاملـة MC1413 وهـي عبـارة عـن مصـفوفة ترانزسـتورات دارلنغتـون قيـادة ملفـات محـرك متعـددة مباشرة من المداخل المنطقية.

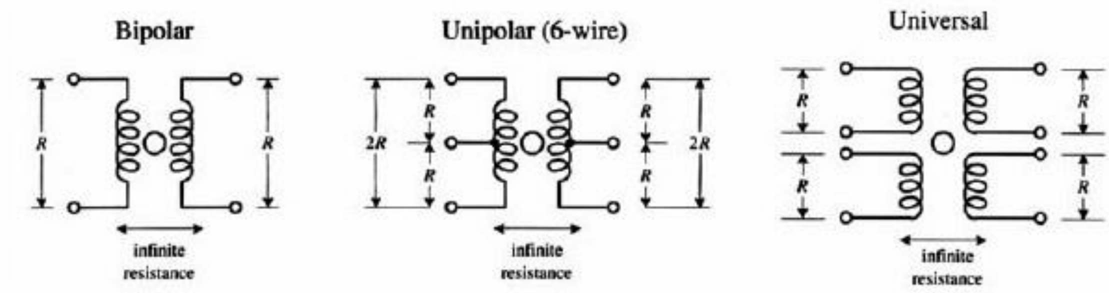
وهناك الدارة L293 ولكنها تستطيع التعامل مع ملفات تحتاج حتى 2A

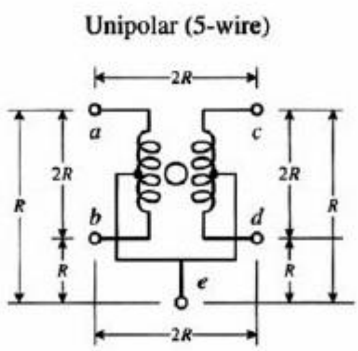
تتعامل الدارة المتكاملة LMDI 8200 مع تيارات حتى 3A وتحوي ديودات حماية داخلية، بعكس الدارات L298 و L293

كلمة أخيرة عن تحديد هوية محركات الخطوة

نقدم فيمـا يلـي الاقتراحـات التاليـة لتحديد مميزات محرك خطـوي غيـر معروف ،إن الأغلبيـة الساحقة من محركات الخطوة المتوفرة في الأسواق هي من الأنواع وحيد القطبية، ثنـائي القطبيـة،

والمحرك العمومي، واعتمادًا على ذلك، إذا كان المحرك الموجود لديك له خمسة أطراف فإنه من النوع وحيد القطبية مع فرعة وسطى مشتركة .أما إذا كان عدد أطراف المحرك ستة فربما يكون المحرك من النوع وحيد القطبية بتفرعات وسطى منفصلة .يمكن وبشكل عام اعتبار المحرك الذي لـه ثمانيـة أطراف محركًا عموميًا .إذا كـان لديك شك بـأن المحرك الموجود لديك من النـوع ذي الممانعة المغناطيسية المتغيرة حاول تدوير الجذع فإذا دار الجذع بحرية فهو على الأغلب من نوع الممانعة المغناطيسية المتغيرة، أما إذا شعرت بمقاومة دوران شبيهة بالمسنن فـإن المحرك يكون من نـوع المغنـاطيس الدائم .بعد أن تحدد نـوع المحرك الـذي لـديك عليك أن تحدد وظـائف الأطراف، ويمكن تحديد الوظائف باستخدام مقياس أوم.



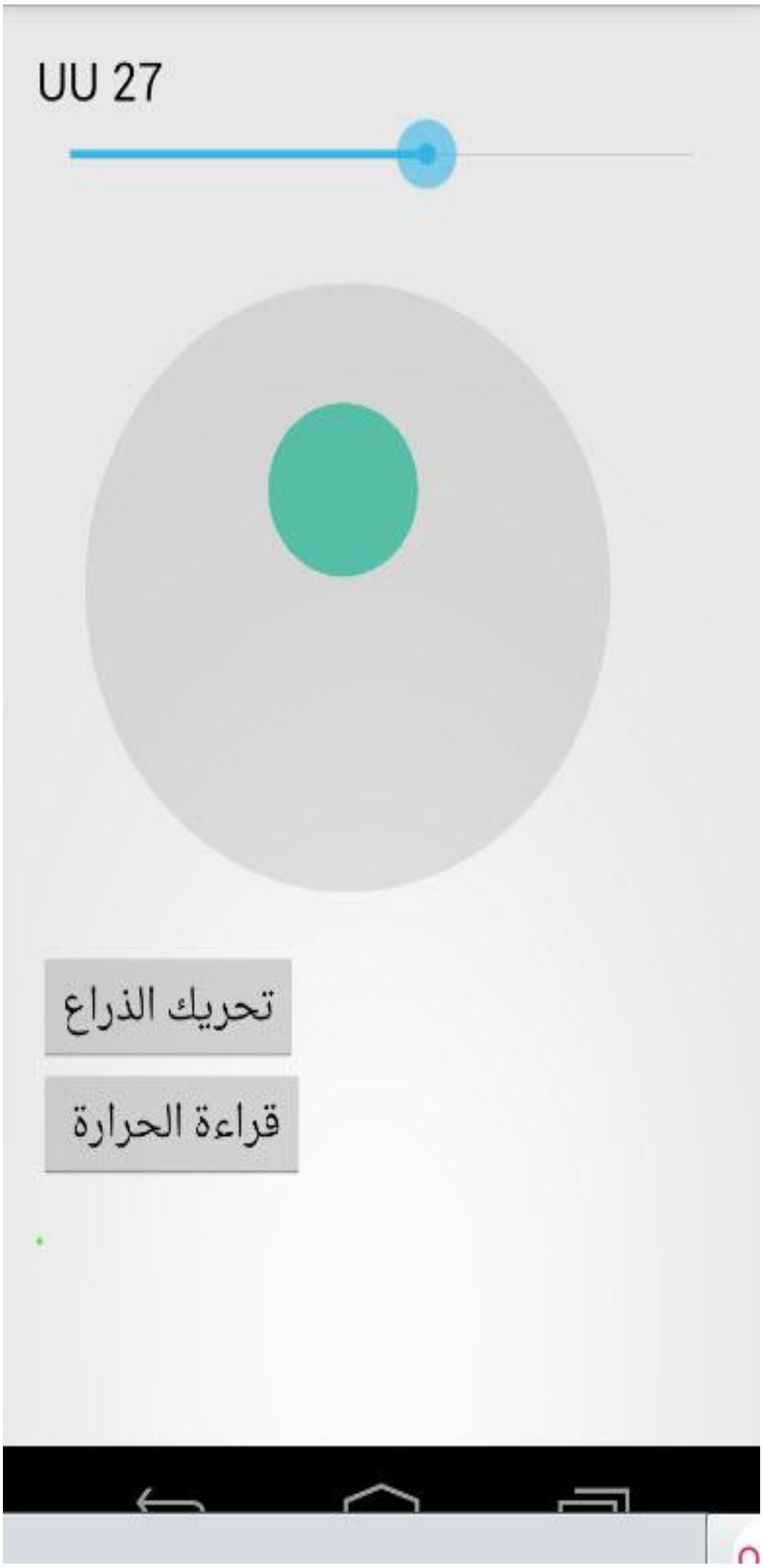


a	b	c	d	e	Resistance
•	•				2R
•		•			2R
•			•		2R
•				•	R
	•	•			2R
	•		•		2R
	•			•	R
		•	•		2R
		•		•	R
			•	•	R

الشكل(26)

إن كشف وظائف أرجل المحرك الخطوي أمر سهل، ويتم ذلك بواسطة مقياس أوم لتحديد زوج الأطراف الذي يعطي مقاومة منخفضة، لأن المقاومة المنخفضة تدل على أن السلكين هما نهايتان لنفس الملف .أما إذا كان الطرفان ليسا نهايتي ملف واحد فإن المقاومة بينهما تكون لا نهائية.

Yasser Code of App RobotControl Joystick For ioio board with ld298 or l293d

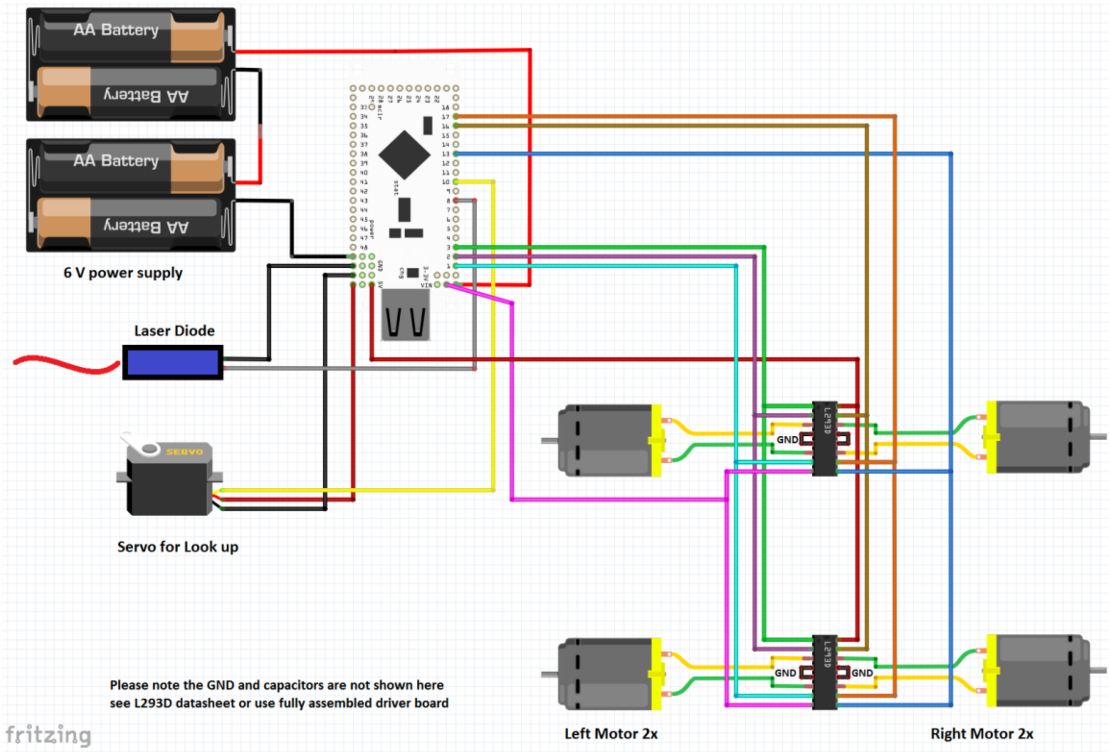


صورة التطبيق على الجهاز

الدارة وتوصيلاتها

Pinout from IOIO to L293d or ld298

- ioio 1 to IN2
- ioio 2 to IN1
- ioio 3 to EN1/2
- ioio 8 to Laser Pointer
- ioio 10 to Digital Servo in
- ioio 13 to EN3/4
- ioio 16 to IN4
- ioio 17 to IN3



كود الأندرويد الشامل

جافا الرئيسي والجويستيك

واجهة

سماحيات وشروط

```
package com.example.robotcontrol;
//YasserKarKouKli  // Java Code
//  الكود البرمجي الملحق المتعامل مع يويو
import ioio.lib.api.DigitalOutput;
import ioio.lib.api.PwmOutput;
import ioio.lib.api.exception.ConnectionLostException;
import ioio.lib.util.AbstractIOIOActivity;
import android.annotation.SuppressLint;
import android.content.Intent;
import android.content.pm.ActivityInfo;
import android.os.Bundle;
import android.view.Display;
import android.view.MotionEvent;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.View.OnTouchListener;
import android.view.Window;
import android.view.WindowManager;
import android.widget.Button;
import android.widget.RelativeLayout;
import android.widget.SeekBar;
import android.widget.SeekBar.OnSeekBarChangeListener;
import android.widget.TextView;
import android.widget.Toast;


@SuppressLint("ClickableViewAccessibility") @SuppressWarnings("deprecation")
public class RobotControll extends AbstractIOIOActivity {

    private SeekBar seekBar_;
    private TextView textView_;
    private String direction_state = "STOP";
    private String strSpeed;
    private Integer jsSpeed;


    //Button buttonUp, buttonDown, buttonLeft, buttonRight;
    //ImageView imageView1;
    //CheckBox cbFlash;
    RelativeLayout layout_joystick;
    JoyStickClass js;
    int screenWidth, screenHeight;
    //Boolean task_state = true;


    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        requestWindowFeature(Window.FEATURE_NO_TITLE);
        getWindow().addFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN
                | WindowManager.LayoutParams.FLAG_KEEP_SCREEN_ON);
        setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_PORTRAIT);
        setContentView(R.layout.activity_robot_controll);
       //


        Toast.makeText(this,"wlecom to ioio world with yasser",3000).show();
        Button servo = (Button) findViewById(R.id.servos);
```

```
        servo.setOnClickListener(new OnClickListener(){

            public void onClick(View arg0) {
                // TODO Auto-generated method stub
                Intent launchIntent = getPackageManager().getLaunchIntentForPackage(
                "ioio.control.servo");
                startActivity(launchIntent);

            }


        });




        Button temp = (Button) findViewById(R.id.button2);
        temp.setOnClickListener(new OnClickListener(){

            public void onClick(View arg0) {
                // TODO Auto-generated method stub
                Intent launchIntent = getPackageManager().getLaunchIntentForPackage(
                "com.ioiobook.templogger");
                startActivity(launchIntent);

            }


        });


        //
        seekBar_ = (SeekBar)findViewById(R.id.seekBar1);
        textView_ = (TextView)findViewById(R.id.textView1);
        textView_.setText("PWM Frequenz: " + String.valueOf(3100 - 650 - seekBar_.getProgress
        ()));

        seekBar_.setOnSeekBarChangeListener(new OnSeekBarChangeListener() {

            @Override
            public void onStopTrackingTouch(SeekBar seekBar) {
            }

            @Override
            public void onStartTrackingTouch(SeekBar seekBar) {
            }

            @Override
            public void onProgressChanged(SeekBar seekBar, int progress,boolean fromUser) {

                textView_.setText("PWM Frequenz: " + String.valueOf(3100 - 650 - seekBar_.
                getProgress()));
            }
        });

        Display display = getWindowManager().getDefaultDisplay();
        screenWidth = display.getWidth();
        screenHeight = display.getHeight();
```

```java
        layout_joystick = (RelativeLayout)findViewById(R.id.layout_joystick);
        js = new JoyStickClass(getApplicationContext(), layout_joystick, R.drawable.
        image_button);
        js.setStickSize(screenHeight / 7, screenHeight / 7);
        js.setLayoutSize(screenHeight / 2, screenHeight / 2);
        js.setLayoutAlpha(50);
        js.setStickAlpha(255);
        //js.setOffset((int)((screenHeight / 9) * 0.6));
        //js.setMinimumDistance((int)((screenHeight / 9) * 0.6));
        js.setOffset((int)(0));
        js.setMinimumDistance((int)(40));

        layout_joystick.setOnTouchListener(new OnTouchListener() {
            long time = System.currentTimeMillis();
            public boolean onTouch(View arg0, MotionEvent arg1) {
                js.drawStick(arg1);
                if(arg1.getAction() == MotionEvent.ACTION_DOWN) {
                    command();
                } else if(arg1.getAction() == MotionEvent.ACTION_MOVE) {
                    if(System.currentTimeMillis() - time > 10) {
                        command();
                        time = System.currentTimeMillis();
                    }
                } else if(arg1.getAction() == MotionEvent.ACTION_UP) {
                    textView_.setText("STOP");
                    direction_state = "STOP";
                    direction_state = "STOP";
                }
                return true;
            }

            public void command() {
                int direction = js.get8Direction();
                int speed = (int)(((js.getDistance() - 40)/3)*2);
                if(speed > 100) speed = 100;
                //speed += 10;
                strSpeed = String.valueOf(speed);
                jsSpeed = speed;

                if(direction == JoyStickClass.STICK_UP) {
                    textView_.setText("UU " + strSpeed);
                    direction_state = "UU";
                } else if(direction == JoyStickClass.STICK_UPRIGHT) {
                    textView_.setText("UR " + strSpeed);
                    direction_state = "UR";
                } else if(direction == JoyStickClass.STICK_RIGHT) {
                    textView_.setText("RR " + strSpeed);
                    direction_state = "RR";
                }  else if(direction == JoyStickClass.STICK_DOWNRIGHT) {
                    textView_.setText("DR " + strSpeed);
                    direction_state = "DR";
                } else if(direction == JoyStickClass.STICK_DOWN) {
                    textView_.setText("DD " + strSpeed);
                    direction_state = "DD";
                } else if(direction == JoyStickClass.STICK_DOWNLEFT) {
                    textView_.setText("DL " + strSpeed);
                    direction_state = "DL";
                } else if(direction == JoyStickClass.STICK_LEFT) {
```

```
                textView_.setText("LL " + strSpeed);
                direction_state = "LL";
            }  else if(direction == JoyStickClass.STICK_UPLEFT) {
                textView_.setText("UL " + strSpeed);
                direction_state = "UL";
            } else if(direction == JoyStickClass.STICK_NONE) {
                textView_.setText("STOP");
                direction_state = "STOP";
                direction_state = "STOP";
            }

        }
    });

}


class IOIOThread extends AbstractIOIOActivity.IOIOThread {


    private DigitalOutput D1A, D1B, D2A, D2B, D3A, D3B, D4A, D4B;
    private PwmOutput PWM1, PWM2, PWM3, PWM4, PWM10, statLed;

    public void setup() throws ConnectionLostException {
        try {

            D1A = ioio_.openDigitalOutput(1, false);
            D1B = ioio_.openDigitalOutput(2, false);
            D2A = ioio_.openDigitalOutput(4, false);
            D2B = ioio_.openDigitalOutput(5, false);
            D3A = ioio_.openDigitalOutput(16,false);
            D3B = ioio_.openDigitalOutput(17,false);
            D4A = ioio_.openDigitalOutput(18,false);
            D4B = ioio_.openDigitalOutput(19,false);

            PWM1 = ioio_.openPwmOutput(3, 100);
            PWM1.setDutyCycle(0);
            PWM2 = ioio_.openPwmOutput(6, 100);
            PWM2.setDutyCycle(0);
            PWM3 = ioio_.openPwmOutput(13, 100);
            PWM3.setDutyCycle(0);
            PWM4 = ioio_.openPwmOutput(14, 100);
            PWM4.setDutyCycle(0);
            PWM10 = ioio_.openPwmOutput(10, 100);

            PWM10.setPulseWidth(3100 - 650 - seekBar_.getProgress());
            statLed = ioio_.openPwmOutput(0, 100);


        } catch (ConnectionLostException e) {
            throw e;

        }
    }


    public void loop() throws ConnectionLostException, InterruptedException {
        try {
```

```
            PWM10.setPulseWidth(3100 - 650 - seekBar_.getProgress());
            statLed.setPulseWidth(seekBar_.getProgress()*6);

            if(direction_state == "UU") {
                PWM1.setDutyCycle((float)jsSpeed / 100);
                PWM2.setDutyCycle((float)jsSpeed / 100);
                PWM3.setDutyCycle((float)jsSpeed / 100);
                PWM4.setDutyCycle((float)jsSpeed / 100);
                D1A.write(true);
                D1B.write(false);
                D2A.write(true);
                D2B.write(false);
                D3A.write(true);
                D3B.write(false);
                D4A.write(true);
                D4B.write(false);
            } else if(direction_state == "DD") {
                PWM1.setDutyCycle((float)jsSpeed / 100);
                PWM2.setDutyCycle((float)jsSpeed / 100);
                PWM3.setDutyCycle((float)jsSpeed / 100);
                PWM4.setDutyCycle((float)jsSpeed / 100);
                D1A.write(false);
                D1B.write(true);
                D2A.write(false);
                D2B.write(true);
                D3A.write(false);
                D3B.write(true);
                D4A.write(false);
                D4B.write(true);
            } else if(direction_state == "LL") {
                PWM1.setDutyCycle((float)jsSpeed / 100);
                PWM2.setDutyCycle((float)jsSpeed / 100);
                PWM3.setDutyCycle((float)jsSpeed / 100);
                PWM4.setDutyCycle((float)jsSpeed / 100);
                D1A.write(false);
                D1B.write(true);
                D2A.write(false);
                D2B.write(true);
                D3A.write(true);
                D3B.write(false);
                D4A.write(true);
                D4B.write(false);
            } else if(direction_state == "RR") {
                PWM1.setDutyCycle((float)jsSpeed / 100);
                PWM2.setDutyCycle((float)jsSpeed / 100);
                PWM3.setDutyCycle((float)jsSpeed / 100);
                PWM4.setDutyCycle((float)jsSpeed / 100);
                D1A.write(true);
                D1B.write(false);
                D2A.write(true);
                D2B.write(false);
                D3A.write(false);
                D3B.write(true);
                D4A.write(false);
                D4B.write(true);
            } else if(direction_state == "UR") {
                PWM1.setDutyCycle((((float)jsSpeed / (float)1.5) + 20) / 100);
```

```
            PWM2.setDutyCycle((((float)jsSpeed / (float)1.5) + 20) / 100);
            PWM3.setDutyCycle((((float)jsSpeed / (float)1.5) - 20) / 100);
            PWM4.setDutyCycle((((float)jsSpeed / (float)1.5) - 20) / 100);
            D1A.write(true);
            D1B.write(false);
            D2A.write(true);
            D2B.write(false);
            D3A.write(true);
            D3B.write(false);
            D4A.write(true);
            D4B.write(false);
        }  else if(direction_state == "UL") {
            PWM1.setDutyCycle((((float)jsSpeed / (float)1.5) - 20) / 100);
            PWM2.setDutyCycle((((float)jsSpeed / (float)1.5) - 20) / 100);
            PWM3.setDutyCycle((((float)jsSpeed / (float)1.5) + 20) / 100);
            PWM4.setDutyCycle((((float)jsSpeed / (float)1.5) + 20) / 100);
            D1A.write(true);
            D1B.write(false);
            D2A.write(true);
            D2B.write(false);
            D3A.write(true);
            D3B.write(false);
            D4A.write(true);
            D4B.write(false);
        } else if(direction_state == "DR") {
            PWM1.setDutyCycle((((float)jsSpeed / (float)1.5) + 20) / 100);
            PWM2.setDutyCycle((((float)jsSpeed / (float)1.5) + 20) / 100);
            PWM3.setDutyCycle((((float)jsSpeed / (float)1.5) - 20) / 100);
            PWM4.setDutyCycle((((float)jsSpeed / (float)1.5) - 20) / 100);
            D1A.write(false);
            D1B.write(true);
            D2A.write(false);
            D2B.write(true);
            D3A.write(false);
            D3B.write(true);
            D4A.write(false);
            D4B.write(true);
        } else if(direction_state == "DL") {
            PWM1.setDutyCycle((((float)jsSpeed / (float)1.5) - 20) / 100);
            PWM2.setDutyCycle((((float)jsSpeed / (float)1.5) - 20) / 100);
            PWM3.setDutyCycle((((float)jsSpeed / (float)1.5) + 20) / 100);
            PWM4.setDutyCycle((((float)jsSpeed / (float)1.5) + 20) / 100);
            D1A.write(false);
            D1B.write(true);
            D2A.write(false);
            D2B.write(true);
            D3A.write(false);
            D3B.write(true);
            D4A.write(false);
            D4B.write(true);
        } else if(direction_state == "STOP") {
            PWM1.setDutyCycle(0);
            PWM2.setDutyCycle(0);
            PWM3.setDutyCycle(0);
            PWM4.setDutyCycle(0);
            D1A.write(false);
            D1B.write(false);
            D2A.write(false);
```

```
                    D2B.write(false);
                    D3A.write(false);
                    D3B.write(false);
                    D4A.write(false);
                    D4B.write(false);
                }


                    sleep(10);

            } catch (ConnectionLostException e) {
                throw e;

            }
        }

    }


    @Override
    protected AbstractIOIOActivity.IOIOThread createIOIOThread() {
        return new IOIOThread();
    }


}
```

```java
package com.example.robotcontrol;

import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Paint;
import android.view.MotionEvent;
import android.view.View;
import android.view.ViewGroup;
import android.view.ViewGroup.LayoutParams;

public class JoyStickClass {
    public static final int STICK_NONE = 0;
    public static final int STICK_UP = 1;
    public static final int STICK_UPRIGHT = 2;
    public static final int STICK_RIGHT = 3;
    public static final int STICK_DOWNRIGHT = 4;
    public static final int STICK_DOWN = 5;
    public static final int STICK_DOWNLEFT = 6;
    public static final int STICK_LEFT = 7;
    public static final int STICK_UPLEFT = 8;

    private int STICK_ALPHA = 200;
    private int LAYOUT_ALPHA = 200;
    private int OFFSET = 0;

    private Context mContext;
    private ViewGroup mLayout;
    private LayoutParams params;
    private int stick_width, stick_height;

    private int position_x = 0, position_y = 0, min_distance = 0;
    private float distance = 0, angle = 0;

    private DrawCanvas draw;
    private Paint paint;
    private Bitmap stick;

    private boolean touch_state = false;

    public JoyStickClass (Context context, ViewGroup layout, int stick_res_id) {
        mContext = context;

        stick = BitmapFactory.decodeResource(mContext.getResources(),
                stick_res_id);

        stick_width = stick.getWidth();
        stick_height = stick.getHeight();

        draw = new DrawCanvas(mContext);
        paint = new Paint();
        mLayout = layout;
        params = mLayout.getLayoutParams();
    }

    public void drawStick(MotionEvent arg1) {
        position_x = (int) (arg1.getX() - (params.width / 2));
```

```
        position_y = (int) (arg1.getY() - (params.height / 2));
        distance = (float) Math.sqrt(Math.pow(position_x, 2) + Math.pow(position_y, 2));
        angle = (float) cal_angle(position_x, position_y);


        if(arg1.getAction() == MotionEvent.ACTION_DOWN) {
            if(distance <= (params.width / 2) - OFFSET) {
                draw.position(arg1.getX(), arg1.getY());
                draw();
                touch_state = true;
            }
        } else if(arg1.getAction() == MotionEvent.ACTION_MOVE && touch_state) {
            if(distance <= (params.width / 2) - OFFSET) {
                draw.position(arg1.getX(), arg1.getY());
                draw();
            } else if(distance > (params.width / 2) - OFFSET){
                float x = (float) (Math.cos(Math.toRadians(cal_angle(position_x, position_y
                ))) * ((params.width / 2) - OFFSET));
                float y = (float) (Math.sin(Math.toRadians(cal_angle(position_x, position_y
                ))) * ((params.height / 2) - OFFSET));
                x += (params.width / 2);
                y += (params.height / 2);
                draw.position(x, y);
                draw();
            } else {
                mLayout.removeView(draw);
            }
        } else if(arg1.getAction() == MotionEvent.ACTION_UP) {
            mLayout.removeView(draw);
            touch_state = false;
        }
    }

    public int[] getPosition() {
        if(distance > min_distance && touch_state) {
            return new int[] { position_x, position_y };
        }
        return new int[] { 0, 0 };
    }

    public int getX() {
        if(distance > min_distance && touch_state) {
            return position_x;
        }
        return 0;
    }

    public int getY() {
        if(distance > min_distance && touch_state) {
            return position_y;
        }
        return 0;
    }

    public float getAngle() {
        if(distance > min_distance && touch_state) {
            return angle;
        }
```

```java
        return 0;
    }

    public float getDistance() {
        if(distance > min_distance && touch_state) {
            return distance;
        }
        return 0;
    }

    public void setMinimumDistance(int minDistance) {
        min_distance = minDistance;
    }

    public int getMinimumDistance() {
        return min_distance;
    }

    public int get8Direction() {
        if(distance > min_distance && touch_state) {
            if(angle >= 247.5 && angle < 292.5 ) {
                return STICK_UP;
            } else if(angle >= 292.5 && angle < 337.5 ) {
                return STICK_UPRIGHT;
            } else if(angle >= 337.5 || angle < 22.5 ) {
                return STICK_RIGHT;
            } else if(angle >= 22.5 && angle < 67.5 ) {
                return STICK_DOWNRIGHT;
            } else if(angle >= 67.5 && angle < 112.5 ) {
                return STICK_DOWN;
            } else if(angle >= 112.5 && angle < 157.5 ) {
                return STICK_DOWNLEFT;
            } else if(angle >= 157.5 && angle < 202.5 ) {
                return STICK_LEFT;
            } else if(angle >= 202.5 && angle < 247.5 ) {
                return STICK_UPLEFT;
            }
        } else if(distance <= min_distance && touch_state) {
            return STICK_NONE;
        }
        return 0;
    }

    public int get4Direction() {
        if(distance > min_distance && touch_state) {
            if(angle >= 225 && angle < 315 ) {
                return STICK_UP;
            } else if(angle >= 315 || angle < 45 ) {
                return STICK_RIGHT;
            } else if(angle >= 45 && angle < 135 ) {
                return STICK_DOWN;
            } else if(angle >= 135 && angle < 225 ) {
                return STICK_LEFT;
            }
        } else if(distance <= min_distance && touch_state) {
            return STICK_NONE;
        }
        return 0;
```

```
    }

    public void setOffset(int offset) {
        OFFSET = offset;
    }

    public int getOffset() {
        return OFFSET;
    }

    public void setStickAlpha(int alpha) {
        STICK_ALPHA = alpha;
        paint.setAlpha(alpha);
    }

    public int getStickAlpha() {
        return STICK_ALPHA;
    }

    public void setLayoutAlpha(int alpha) {
        LAYOUT_ALPHA = alpha;
        mLayout.getBackground().setAlpha(alpha);
    }

    public int getLayoutAlpha() {
        return LAYOUT_ALPHA;
    }

    public void setStickSize(int width, int height) {
        stick = Bitmap.createScaledBitmap(stick, width, height, false);
        stick_width = stick.getWidth();
        stick_height = stick.getHeight();
    }

    public void setStickWidth(int width) {
        stick = Bitmap.createScaledBitmap(stick, width, stick_height, false);
        stick_width = stick.getWidth();
    }

    public void setStickHeight(int height) {
        stick = Bitmap.createScaledBitmap(stick, stick_width, height, false);
        stick_height = stick.getHeight();
    }

    public int getStickWidth() {
        return stick_width;
    }

    public int getStickHeight() {
        return stick_height;
    }

    public void setLayoutSize(int width, int height) {
        params.width = width;
        params.height = height;
    }

    public int getLayoutWidth() {
```

```
        return params.width;
    }

    public int getLayoutHeight() {
        return params.height;
    }

    private double cal_angle(float x, float y) {
        if(x >= 0 && y >= 0)
            return Math.toDegrees(Math.atan(y / x));
        else if(x < 0 && y >= 0)
            return Math.toDegrees(Math.atan(y / x)) + 180;
        else if(x < 0 && y < 0)
            return Math.toDegrees(Math.atan(y / x)) + 180;
        else if(x >= 0 && y < 0)
            return Math.toDegrees(Math.atan(y / x)) + 360;
        return 0;
    }

    private void draw() {
        try {
            mLayout.removeView(draw);
        } catch (Exception e) { }
        mLayout.addView(draw);
    }

    private class DrawCanvas extends View{
        float x, y;

        private DrawCanvas(Context mContext) {
             super(mContext);
         }

         public void onDraw(Canvas canvas) {
             canvas.drawBitmap(stick, x, y, paint);
         }

         private void position(float pos_x, float pos_y) {
            x = pos_x - (stick_width / 2);
            y = pos_y - (stick_height / 2);
         }
     }
}
```

```
// layout Code of Robot control' yasser karkoukli
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/LinearLayout1"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="left"
    android:orientation="vertical"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".RobotControll" >

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"

        android:text="@string/large_text"
        android:textAppearance="?android:attr/textAppearanceLarge" />

    <SeekBar
        android:id="@+id/seekBar1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"

        android:max="1900"
        android:progress="850" />

    <RelativeLayout
        android:id="@+id/layout_joystick"
        android:layout_width="150dp"
        android:layout_height="150dp"
        android:layout_marginTop="14dp"
        android:background="@drawable/image_button_bg" >
    </RelativeLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="right" >

    </LinearLayout>

    <Button
        android:id="@+id/servos"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/servo" />

    <Button
        android:id="@+id/button2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/temp" />

    <ImageView
```

```
        android:id="@+id/imageView1"
        android:layout_width="3dp"
        android:layout_height="wrap_content"
        android:src="@drawable/ic_launcher" />

</LinearLayout>
```

```
//كود السماحيات
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.robotcontrol"
    android:versionCode="1"
    android:versionName="1.0" android:installLocation="auto">

    <uses-sdk
        android:minSdkVersion="10"
        android:targetSdkVersion="19" />
    <uses-permission android:name="android.permission.BLUETOOTH"/>

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.example.robotcontrol.RobotControll"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Control by smart Phone systems

Yasser karkoukli

